

FOR THE

MIDDLE DISTRICT OF PENNSYLVANIA

CIVIL ACTION FILE NO

BALLY MANUFACTURING CORPORATION,

No.

D. GOTTLIEB & CO., WILLIAMS ELECTRONICS,
INC. and ROCKWELL INTERNATIONAL CORPORATION,

To Mr. Gregory Cox
683 Berkshire Drive
State College, PA 16801

YOU ARE HEREBY COMMANDED to appear in the United States District Court for the
Middle District of Pennsylvania
at Miller, Kistler & Campbell in the city of State College on
the 1500 So. Atherton Street 1981 at 9:30 o'clock A. M. to
the first day of September,
testify on behalf of defendants, D. Gottlieb & Co., Williams Electronics, Inc.
and Rockwell International Corporation
in the above entitled action and bring with you the documents set forth in the
attachment to this Subpoena.

August 25, 1981

Attorney for Defendants
1500 So. Atherton Street
State College, PA 16801

DONALD R. [Signature]

By

Clerk.

Deputy Clerk.

RETURN ON SERVICE

Received this subpoena at _____ on
and on _____ at _____
served it on the within named _____
by delivering a copy to h _____ and tendering to h _____ the fee for one day's attendance and the mileage
allowed by law.¹

Dated:

_____, 19____

By _____

Service Fees

Travel _____ \$

Services _____

Total _____ \$

Subscribed and sworn to before me, a

this

, 19

Greg Cox
X-1
LVB
211

Notary Seal

ATTACHMENT TO SUBPOENA OF MR. GREGORY COX

You are requested to produce for inspection and copying in connection with your deposition all documents and things in your possession, custody or control relating to the subject matter of the Affidavit and Declaration which you executed March 5, 1981 and July 9, 1981, respectively, including without limitation:

1. Documents and things relating to your employment by Cyan Engineering during 1974 which refer or relate to the projects you worked on or the people you worked with.
2. Documents relating to El Toro pinball machine project, the computer program you developed for the El Toro prototype or the Cyan open house in June, 1974.
3. Documents relating to your contacts and discussions with Intel personnel and/or Steven Mayer.
4. Correspondence to or from Bally or its attorneys relating to any of the foregoing matters and specifically including, without limitation, correspondence, notes, tapes, etc., relating to the preparation of the Affidavit and Declaration.

United States District Court

FOR THE

MIDDLE DISTRICT OF PENNSYLVANIA

BALLY MANUFACTURING CORPORATION,

CIVIL ACTION FILE NO.

vs.

D. GOTTLIEB & CO., WILLIAMS ELECTRONICS,
INC. and ROCKWELL INTERNATIONAL CORPORATION,

No.

To Mr. Gregory Cox
683 Berkshire Drive
State College, PA 16801

YOU ARE HEREBY COMMANDED to appear in the United States District Court for the
Middle District of Pennsylvania
at Miller, Kistler & Campbell in the city of State College on
the 1500 So. Atherton Street 1981 at 9:30 o'clock A. M. to
the first day of September,
testify on behalf of defendants, D. Gottlieb & Co., Williams Electronics, Inc.
and Rockwell International Corporation
in the above entitled action and bring with you the documents set forth in the
attachment to this Subpoena.

August 25, 1981
Eugene Williams
Attorney for Defendants
1500 So. Atherton Street
State College, PA 16801

DONALD R. BERRY
By *Sylvia Longchoteky*
Deputy Clerk.

RETURN ON SERVICE

Received this subpoena at _____ on _____
and on _____ at _____
served it on the within named _____
by delivering a copy to h _____ and tendering to h _____ the fee for one day's attendance and the mileage
allowed by law.¹

Dated: _____

By _____

Service Fees
Travel _____ \$
Services _____

Total _____ \$

Subscribed and sworn to before me, a
day _____, 19 _____

this

ATTACHMENT TO SUBPOENA OF MR. GREGORY COX

You are requested to produce for inspection and copying in connection with your deposition all documents and things in your possession, custody or control relating to the subject matter of the Affidavit and Declaration which you executed March 5, 1981 and July 9, 1981, respectively, including without limitation:

1. Documents and things relating to your employment by Cyan Engineering during 1974 which refer or relate to the projects you worked on or the people you worked with.

2. Documents relating to El Toro pinball machine project, the computer program you developed for the El Toro prototype or the Cyan open house in June, 1974.

3. Documents relating to your contacts and discussions with Intel personnel and/or Steven Mayer.

4. Correspondence to or from Bally or its attorneys relating to any of the foregoing matters and specifically including, without limitation, correspondence, notes, tapes, etc., relating to the preparation of the Affidavit and Declaration.

United States District Court

FOR THE

BALLY MANUFACTURING CORPORATION,

CIVIL ACTION FILE NO.

vs.

No.

D. GOTTLIEB & CO., WILLIAMS ELECTRONICS, INC.
and ROCKWELL INTERNATIONAL CORPORATIONTo Mr. Gregory Cox
683 Berkshire Drive
State College, PA 16801

YOU ARE HEREBY COMMANDED to appear in the United States District Court for the
 at Miller, Kistler & Campbell Middle District of Pennsylvania on
 the 1500 S. Atherton Street in the city of State College
the 11th day of September, 1981 at 9:30 o'clock A. M. to
 testify on behalf of defendants, D. Gottlieb & Co., Williams Electronics,
 Inc. and Rockwell International Corporation
 in the above entitled action and bring with you the documents set forth in the
 attachment to this Subpoena.

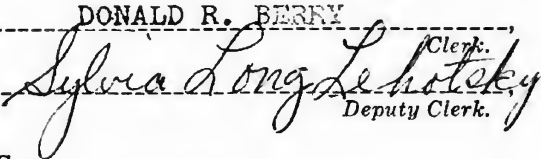
September 8, 1981


 Attorney for Defendants
1500 S. Atherton Street
Address

State College, PA 16801

DONALD R. BERRY

By


 Deputy Clerk.

RETURN ON SERVICE

Received this subpoena at Miller, Kistler & Campbell, Inc., office on September 8,
 1981 and on Sept. 9, 1981 at 412 S. Allen Street, State College, Pa.
 served it on the within named Gregory Cox by and through his attorney Benjamin Novak, Esq.,
 by delivering a copy to him and tendering to him the fee for one day's attendance and the mileage
 allowed by law.¹

Dated:

Sept. 9, 1981

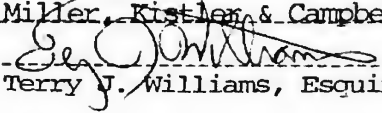
Service Fees

Travel \$
Services

Total \$

Miller, Kistler & Campbell, Inc.,

By


 Terry J. Williams, Esquire

Subscribed and sworn to before me, a Notary Public, this

day of

September

, 1981.

ATTACHMENT TO SUBPOENA OF MR. GREGORY COX

You are requested to produce for inspection and copying in connection with your deposition all documents and things in your possession, custody or control relating to the subject matter of the Affidavit and Declaration which you executed March 5, 1981 and July 9, 1981, respectively, including without limitation:

1. Documents and things relating to your employment by Cyan Engineering during 1974 which refer or relate to the projects you worked on or the people you worked with.

2. Documents referring or relating to the El Toro pinball machine project, the computer program you developed for the El Toro prototype or the Cyan open house in June, 1974.

3. Documents referring or relating to your contacts and discussions with Intel personnel and/or Steven Mayer.

4. Correspondence to or from Bally or its attorneys relating to any of the foregoing matters and specifically including, without limitation, correspondence, notes, tapes, etc., relating to the preparation of the Affidavit and Declaration.

IN THE UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF ILLINOIS
EASTERN DIVISION

BALLY MANUFACTURING CORPORATION,

Plaintiff,

vs.

D. GOTTLIEB & CO., WILLIAMS
ELECTRONICS, INC. and ROCKWELL
INTERNATIONAL CORPORATION,

Defendants.

CIVIL ACTION NO.

78-C-2246

NOTICE OF DEPOSITION

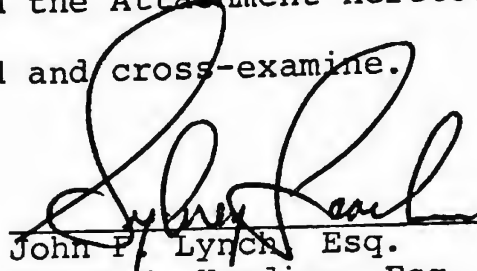
To: Donald L. Welsh
A. Sidney Katz
Fitch, Even, Tabin, Flannery & Welsh
135 S. LaSalle Street
Chicago, IL 60603

PLEASE TAKE NOTICE that at 9:30 a.m. on the 11th day of September, 1981, at the offices of Miller, Kistler & Cambell, Inc., 1500 South Atherton Street, State College, Pennsylvania 16801, or such other place agreed to by counsel for the parties, the defendants in the above-captioned action will take the deposition of Mr. Gregory Cox whose post office address is 683 Berkshire Drive, State College, Pennsylvania 16801, upon oral examination pursuant to the Federal Rules of Civil Procedure before a Notary Public or other officer authorized by law to administer oaths. The oral examination

Greg Cox X-4
LvB
9/11/81

will continue from day to day or until completed or until adjourned by defendants. The witness will be asked to produce the items set forth in the Attachment hereto.

You are invited to attend and cross-examine.



John P. Lynch, Esq.
Wayne M. Harding, Esq.
Sydney M. Leach, Esq.
ARNOLD, WHITE & DURKEE
P. O. Box 4433
Houston, TX 77210

OF COUNSEL:

Melvin M. Goldenberg
William T. Rifkin
McDougall, Hersh & Scott
135 S. LaSalle Street
Chicago, IL 60603

Attorneys for Williams
Electronics, Inc.

Attorneys for D. Gottlieb & Co.
and Rockwell International
Corporation

ATTACHMENT TO SUBPOENA OF MR. GREGORY COX

You are requested to produce for inspection and copying in connection with your deposition all documents and things in your possession, custody or control relating to the subject matter of the Affidavit and Declaration which you executed March 5, 1981 and July 9, 1981, respectively, including without limitation:

1. Documents and things relating to your employment by Cyan Engineering during 1974 which refer or relate to the projects you worked on or the people you worked with.

2. Documents referring or relating to the El Toro pinball machine project, the computer program you developed for the El Toro prototype or the Cyan open house in June, 1974.

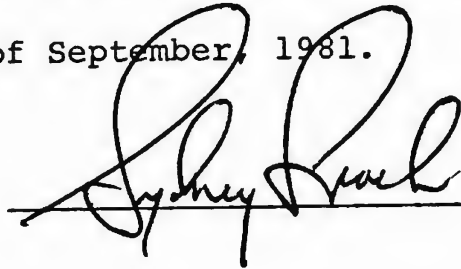
3. Documents referring or relating to your contacts and discussions with Intel personnel and/or Steven Mayer.

4. Correspondence to or from Bally or its attorneys relating to any of the foregoing matters and specifically including, without limitation, correspondence, notes, tapes, etc., relating to the preparation of the Affidavit and Declaration.



CERTIFICATE OF SERVICE

I hereby certify that a copy of the foregoing NOTICE OF DEPOSITION was served on plaintiff by depositing a copy with the United States Postal Service, first class mail, postage prepaid to Donald L. Welsh, A. Sidney Katz, Fitch, Even, Tabin, Flannery & Welsh, 135 S. LaSalle Street, Chicago, Illinois 60603 this 1st day of September, 1981.



How to activate
multiple players

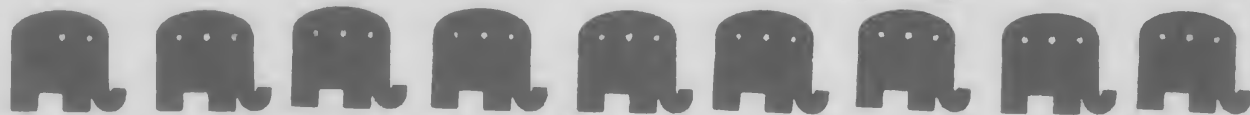
Retained by Scheneyer's

Co. - Consult them

requiring

~~Retain~~ as technical
Consultant / specialist

Greg Cox X-5
LVB
9/11/81



Elect

CANNICI - VAN NATTA
For Council

..... They've earned the right to serve.....

Paid for by the Committee to elect Cannici - Van Natta.



Started Cyan

4 March 1974

Terminated

16 August 1974

Statements relating to
players critical - disjunct
& explain - # of players it could
handle - how multiple scores
display

Greg Cox X-6
LVB
9/11/81

Pinball patent

as a friend of
Article #6

Steve Mayer

June '78 - work done '73-'74

Consulting #50/hr.

Jerry Schnayer 312-372-7842

Fitch Even Et al,

Chicago

Represent Bally manuf.

Steve Mayer

A Cole 209-714

916

Cyan Engineering

916 - 273-6194

408 745-5077

Greg Cox X-7
LVB
9/11/51

Determine start date Cyan Eng. ^{occupied house} 2/28/74
restart date at LVB ^{signed for agreement} 8/19/74
i.e. report dates at Cyan Eng. ^{started for UE} 8/23/74

Look for responses stuck switches.
multiple closures.

Greg Cox X-8
LVB
9/11/81

Accounting Charges

Wednesday 2/18
 telephone conference 1/2 hr.
 Dinner Conference 3 1/2 hr.
 Friday 2/20
 telephone conference 1/2 hour
 Monday 2/23
 telephone conf 1/2 hr.
 Dinner conf 3 1/2 hr.
Billed 2/28 Tues 2/24 3 1/2 hr.

Wed 3/4 Dinner Conf 2 hr.

Thur. 3/5 Conf. 1/2 hr.

Billed 4/3/81

4/3 Prog. Analysis 1 1/2 hr.
 4/7 Dinner Conf 4 1/2 hr.
 later 7/5 hr.
 7/7 4 hrs
 7/7 1/2 hr

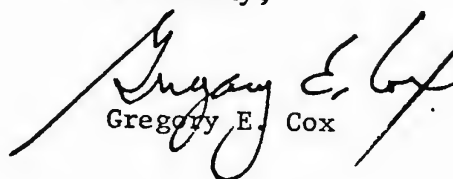
Greg Cox X-9
 LVB
 9/11/81

Fitch, Even, Tabin, Flannery & Welsh
Suite 900
135 South LaSalle Street
Chicago, Illinois 60603

Attention: Mr. Jerry Schnayer

Enclosed please find my bill for consulting services regarding the
Atari EL TORO pinball game for the period 15-28 February 1981.

Sincerely,


Gregory E. Cox

Greg Cox X-10
LVB
9/11/81

Statement

From Gregory E. Cox

152 Buena Vista Dr. Ringwood, N.J. 07456

March 2 1981

To Fitch, Even, Tabin, Flannery & Welsh

Street Suite 900 - 135 South LaSalle Street

City Chicago, Ill. 60603 Terms 30 Days Net

Billing for consulting services 15-28 Feb. 1981

Date	Service	Hours	Fee
2/18	Telephone conference	0.5	\$25.00
	Dinner conference	3.5	175.00
2/20	Telephone conference	0.5	25.00
2/23	Telephone conference	0.5	25.00
	EL TORO program analysis	3.25	162.50
2/24	EL TORO program analysis	3.5	175.00
	Total	11.75	\$587.50
	Billing rate = \$50./hr		

Statement

From GREGORY COX

152 BUENA VISTA DR.

RIDGEWOOD APTS. 07456 5/4 19 81

To FITCH, EVAN et. al.

Street SUITE 900 - 135 S. LA SALLE ST.

City CHICAGO, ILL.

Terms 30 DAYS NET

BILL FOR CONSULTING SERVICES

4/1 TO 4/30/1981

\$300.

BILLING RATE: \$50/hr.

Greg Cox X-11
LVB
9/1/81

4 May 1981

Fitch, Even, Fabin, Flannery & Welch

Suite 900

135 South La Salle Street

Chicago, Illinois 60603

Attn: Mr. J. Schnayer

Enclosed please find bill for April
consulting services. Things are going well
at the new job but still no letter on the
house. I hope you get a chance to visit State
College soon, springtime is very nice here.

Yours very truly,
Gregory Cox

Greg Cox X-12

lub

9/11/81

Statement

From GREGORY COX
683 BERKSHIRE DR.
STATE COLLEGE, PA 16801 9/1/81
 To FITCH, EVEN, et. al.
 Street SUITE 900 135 S. LA SALLE ST.
 City CHICAGO, ILL. 60603 Terms 30 DAYS NET

BILL FOR CONSULTING SERVICES

8/1 TO 8/31 1981			
CONSULTATION (2 HOURS @ \$75/HR)			\$150.
LEGAL FEES			750.
TOTAL			\$900.

Greg Cox X-13
 LVB
 9/11/81

MC
7/9/81

DECLARATION FOR GREGORY COX

I, Gregory Cox, hereby declare:

1. I currently reside at 683 Berkshire Dr., State College, Pennsylvania.

2. On March 5, 1981 I executed an affidavit concerning my activities while employed by Cyan Engineering in 1974.

3. For about the first six weeks of my employment at Cyan Engineering Steven Mayer and I had ongoing discussions with Intel employees and representatives, including Intel applications engineers, concerning our work on the El Toro project. These discussions included several telephone calls which I had with Intel applications engineers referred to me by Steven Mayer. A majority of the discussions involved telephone conversations between Steven Mayer and Intel. I became aware of the details of the telephone conversations between Steven Mayer and Intel through conversations with him where Steven Mayer discussed with me the information he had obtained from Intel. In these discussions with Intel, Steven Mayer and I asked technical questions about the operation of Intel microprocessor related products including the MCS-4 microcomputer chip set and the Intellec development system, and how to interface them to components of the El Toro and other games. Additionally, such discussions with Intel included questions by us about how to program the Intellec development system to control the various components of the El Toro and other games. Steven Mayer and I used the information that we learned from these discussions with Intel in designing control circuits for the El Toro prototype.

4. While working at Cyan Engineering in 1974 I generated and developed the software program for the El Toro prototype. Attached hereto

Greg Cox X-14
LVB
9/11/81

and the like so much

as Exhibits A and B are copies of documents labeled respectively GD54 and GD55 which I generated in connection with this work. GD54 is a preliminary version of the software program which I designed, coded and put into operation in the El Toro prototype. GD55 is a patch log which I generated after debugging GD54. GD54 and GD55 taken together, or their exact functional equivalents, comprise a version of the software program for the El Toro prototype which I generated and which was used with the El Toro prototype at the "open house" referred to in paragraph 13 of my above mentioned affidavit (hereinafter referred to as the "open house" version of the software program).

BC 7/9/81

5. I participated in developing the software program for the El Toro prototype. During this development I observed a switch debounce problem with the playfield switches which caused a single switch closure to register multiple scores. In order to correct this switch debounce problem I designed into the software program for the El Toro prototype switch debouncing software program instructions which are included in the software program documented in GD54 and GD55, and which were contained in the "open house" version of the software program. I designed the switch debouncing software program instructions so that the El Toro would not react to successive switch closures until approximately 35 milliseconds after all closed switches returned to an open state. As a result of these switch debouncing software program instructions, in the case of a continuous closure of a playfield switch, such as a "stuck switch", the El Toro would not respond to other switch closures. Therefore, if a stuck switch occurred the digital displays of the El Toro would not increase the score which normally would have resulted from closure of switches other than the stuck switch, the solenoids such as the kickers would not activate, and the lamps would remain in the state they were in prior to the occurrence of the stuck switch.

BC 7/9/81

6. During my participation in the design and development of the El Toro prototype I became familiar with the Atari and Cyan Engineering personnel who worked on the project, which included Steven Mayer, Larry Emmons, Mike Rogers, Ed Scheetler, Joel Miller and others. Particularly, I had conversations with these personnel wherein they told me about their past experiences and background relating to pinball machines. Through these conversations I became aware that none of these personnel had any direct experience in the design and/or manufacture of pinball machines.

7. Shortly after being hired by Cyan Engineering, Steven Mayer and/or Larry Emmons discussed with me the strictly enforced company policy at Cyan Engineering and Atari of not discussing any aspect of our design activities with anybody except Cyan Engineering and Atari employees.

Steven Mayer and/or Larry Emmons told me that the reason for this policy was as follows. Because of the competitive nature of the game business, where most of the sales of newly released games occurred within 6 months of their initial release to the market place, if the design of a new game were stolen, copied and sold by a competitor prior to its being placed on the market by Atari, due to the short sales life of games Atari would realize a significant loss of sales.

AC 7/9/81
AC 7/9/81

8. Through discussions with employees of Cyan Engineering and Atari it is my understanding that all employees were aware of this policy referred to in the proceeding above paragraph which was strictly enforced throughout the entire period time I was employed with Cyan Engineering in 1974.

AC 7/9/81

9. Consistent with the policy referred in the preceding two paragraphs above throughout my entire employment at Cyan Engineering all drawings relating to design efforts at Cyan Engineering when not in use were locked in security cabinets, the Cyan Engineering facility had a special burgular alarm system, and guards patroled the facility during nonbusiness hours.

10. Prior to the "open house" referred to in paragraph 13 of my above mentioned affidavit I participated in discussions with Cyan Engineering personnel, including Steven Mayer and Larry Emmons, where I was told that the purpose of the "open house" was to allow Atari personnel to view our facilities and the projects that we were working on, and to set up a social gathering of Atari and Cyan Engineering employees and their families to foster a closer working relationship between the two parts of the company. During these discussions I was also told some of the Atari employees who would be present at the "open house".

BC
7/9/81

11. It is my understanding that the only people present at the "open house" were Cyan Engineering and Atari employees, and members of their immediate families. My understanding is based on the fact that, with respect to every person who attended the "open house", I either personally knew them or they were introduced to me. My understanding is also based on the discussion referred to in the preceding paragraph above.

BC
7/9/81

12. It is my understanding that everyone present at the "open house" understood that all information learned at the "open house" was to be kept strictly company confidential. My understanding is based on my knowledge of the strictly enforced company policy as discussed in paragraph 7, 8 and 9 above, and on my understanding that all attendees at the "open house" were either company employees or members of their immediate family who would be fully aware of these policies, and the requirement to keep all information concerning company work company confidential.

BC
7/9/81

The undersigned declarant, Gregory Cox, declares further that all statements made herein of his own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false

statements and the like so made are punishable by fine or imprisonment,
or both, under section 1001 of Title 18 of the United States Code.

Declarant's full name:

Gregory Cox
Gregory Cox

Date

7/9/81

Post office address:

683 BERKSHIRE DR.
STATE COLLEGE, PA. 16801

Exhibit #1

139	26	JUN	43
140	8C	832	40
832	340	LD 14	AE
3	1	JAZ	14
4	2	++13	4E
5	3	Lm 0	DO
6	4	JMS	50
7	5	IXOP + 2	FO
8	6	LDM 14	DE
9	7	JMS	52
840	8	RPOOP	11
1	9	LDM 0	DO
2	A	JMS	52
3	B	RPOOP	11
4	C	JMS	50
5	D	IXOP + 1	EF
6	E	LD 12	AE
7	F	JMS	52
848	350	COOP	00
849	351	JUN	40
850	352	141	FD

LD 11

20	14	NOP	00
21	15	"	00
22	16	"	00
23	17	"	00
24	18	"	00
25	19	"	00

142	1C	JUN	43
143	1F	832	40
139	8E	JMS	52
140	8C	COOP	00

846	34E	LD 11	AE
850	352	144	90

848 54
 1-11-79
 J.A. P., Notary

NOTE: To Accommodate COK
CHANGE: 265 → NOP
266 → NOP

Added 51 Instr.

```

0: VECTR = 258
0:
0: / PINBALL EXECUTIVE
0: /
0: LDM 12 /SET GAME OVER AND BACKGROUND ON
1: XCH 2
2: PINBAL LDM 0 /TURN LEDS OFF
3: JMS RPOOP
5: LDM 6 /POSITION X TO C LDM 0 /POSITION X TO A
6: JMS IXOP+2
8: WRR
9: JTZ * /WAIT FOR 120 HZ PULSE
11: LD 14 /ENABLE FLIPPERS IF BALL COUNT > 0
12: JAZ **5
14: LDM 13
15: JMS RPOOP
17: LDM 1 /TEST AND SET PHASE
18: WRR
19: CLB
20: JTZ **3
22: IAC
23: XCH 4
24: FIM 1P,8 /RESET X,Y
26: ISZ 2, * /DELAY TO SYNCH SCR TO LIGHTS
28: LD 4 /TEST PHASE
29: RAR
30: LD 7
31: JCO **3
33: LD 9
34: WRR /OUTPUT FIRST 4 LIGHTS
35: LDM 15
36: JMS RPOOP /ENABLE ROM 0
38: LD 6
39: JCO **3
41: LD 8
42: WRR /OUTPUT LAST 4 LIGHTS
43: LDM 14 LDM 13
44: JMS RPOOP /ENABLE ROM 0
46: JMS IXOP+1 /POSITION X TO A
48: LD 13
49: CLC
50: DAA
51: JMS RPOOP /TURN ON LED A SCORE 10,000
53: JMS DLY /DELAY FOR LIGHT
55: LDM 5 /TEST OUT HOLE
56: WRR
57: JTZ PB1
59: CLB /RESET RA1-3,GTB
60: XCH 0
61: LD 14 /TEST BALL COUNT
62: JAZ OH1
64: LD 8
65: RAR
66: JAZ **6 /TEST SHOOT AGAIN
68: INC 14 /INCREMENT BALL COUNT

```

LDM 15 /set Pos Count
XCH 5
FIM 0P,16 /select Rom 1 for I/O
SRC 0P

JAZ **8
LDM 14

LDM 0
JMS RPOOP
JUN **6
LDM 8
XCH 0
ISZ 0, *

LDM 15 /SYNC TO SCR
XCH 0
ISZ 0, *

LDM 0
JMS RPOOP

LDM 0
JMS RPOOP
WRR /WAIT FOR HIT COK
JTZ *
LDM 9 /POSITION X TO J
JMS IXOP+2

LDM 6 - /TEST SHOOT
WRR
JTZ PH1

JAZ PB1
LDM 5
WRR


```

70: RAL
71: XCH 8
72: LD 14 /DECREMENT BALL COUNT
73: DAC
74: XCH 14
75: JAZ OH3
77: OH2 JMS SOLEN /OUTPUT OUT HOLE KICKER
79: XCH 0
80: JTN *-1
82: ISZ 0, *-2
84: JUN PINBAL
86: OH3 LDM 4 /TEST MATCH ENABLE
87: WRR
88: JTZ SGO
90: JMS MATCH /GET MATCH VALUE
92: CLC
93: LD 0 /TEST IF A MATCH
94: SUB 10
95: JAN SGO
97: JMS GMINC /INCREMENT GAME COUNT
99: SGO LD 8 /SET GAME OVER LIGHT ON
100: JMS SB2
102: LD 0
103: XCH 8
104: JUN PBI /CONTINUE PROCESSING
106: OH1 LDM 5 /TEST START SWITCH
107: WRR
108: JTZ PBI
110: LD 15 /START A NEW GAME
111: JAZ PBI
113: DAC
114: XCH 15 /DECREMENT GAMES REMAINING
115: LDM 7 /TEST 5BALL SWITCH
116: WRR
117: LDM 3
118: JTZ *+3
120: LDM 5
121: XCH 14 /SET BALLS REMAINING
122: FIM 3P,0 /RESET REGISTER DATA
124: FIM 4P,128
126: FIM 5P,0
128: FIM 6P,0
130: LDM 5 /SET ROM PORT TO OUT HOLE KICKER
131: WRR
132: JUN OH2
134: PBI CLB /GET ON WITH IT
135: XCH 4 /RESET HIT COUNT
136: JMS LXOP /POSITION X TO B
138: LD 12
139: JMS LOOP /OUTPUT B LED TEST SWITCHES INCREMENT X
141: LD 11
142: JMS LOOP /OUTPUT LED C TEST SWITCHES INCREMENT X
144: LD 10
145: JMS LOOP /OUTPUT LED D TEST SWITCHES INCREMENT X
147: LD 14
148: JMS RLOOP /OUTPUT LED E
150: WRR
151: LD 4 /TEST HIT COUNT
152: JAZ *+6
154: CLB /RESET MISS COUNT
155: XCH 5

```

LD 14

JTN *

PH1 LD 15 /START A NEW GAME

FIM 4P,128

164: LD 8 /TURN BACKGROUND OFF

165: RAL

166: CLC

167: RAR

168: XCH 8

169: CLB

170: JUN *+11

172: LDM 1 /TEST TILT

173: WRR

174: JTZ *+11

176: LD 8 /TURN TILT ON

177: JMS SBI

179: LD 0

180: XCH 8

181: CLB /SET BALL COUNT = 0

182: XCH 14

183: JUN SGO /SET GAME OVER

185: LDM 6 /TEST 1P

186: WRR

187: JTZ *+7

189: JMS GMINC /INCREMENT GAME COUNT

191: CLB

192: JUN OR2+2

194: LDM 7 /TEST 3P

195: WRR

196: JTZ *+8

198: JMS GMINC /INCREMENT GAMES REMAINING BY 3

200: JMS GMINC

202: JUN *-13

204: JMS DLY /DELAY FOR LIGHTS

206: JMS IXOP /POSITION X TO F

208: LD 15

209: CLC

210: DAA

211: XCH 4

212: TCC

213: JMS RPOOP /OUTPUT LED F GAMES REMAINING 10

215: JMS DLY /DELAY FOR LIGHTS

217: JMS IXOP /POSITION X TO G

219: LD 4

220: JMS RPOOP /OUTPUT LED G GAMES REMAINING 1

222: JMS DLY /DELAY FOR LIGHTS

224: ~~JMS IXOP /POSITION X TO H~~

226: LD 14 /TEST BALL COUNT FOR MATCH

227: ~~JAZ *+9~~

229: JMS MATCH

231: LD 0

232: JMS RPOOP /OUTPUT LED H MATCH TENS

234: JMS DLY /DELAY FOR LIGHTS

236: JUN PINBAL /GO ROUND AGAIN

238: / X COORDINATE OUTPUT ROUTINE

238:/

238: IXOP INC 2 /ENTRY HERE FOR X COORD INCREMENT

239: LD 2

240: WRR /ENTRY HERE FOR X COORD IN ACC

241: LDM 1

242: DCL

243: JUN URET

245: / KNOCKER AND CHIME OUTPUT ROUTINE

245:/

245: XCH WRR /OUTPUT ACC TO DECODER

246: LDM 2

247: DCL /CLOCK DECODER

248: JUN URET

250: /

JT2

*+6

JT0

*

JMS

GMINC

LDM

7 /TEST 3P

WRR

JT2

*+10

JT0

*

JMS

GMINC

JMS

GMINC

JMS

GMINC

LDM 8

JMS IXOP+2

POSITION X TO I

JAN *+7

JMS IXOP


```

250:GMINC ISZ 15, **4 /INCREMENT GAME COUNT
252: LDM 15
253: XCH 15
254: LDM 3 /OUTPUT KNOCKER
255: JMS KCH
257: BBL 0
258: =512
512:/
512:/ SWITCH TEST LOGIC LOOP
512:/
512:LOOP JMS RPOOP,
514: LD 3
515: WRR
516: JTZ **4 /TEST SWITCH
518: JUN VECTR /VECTOR TO SERVICE ROUTINE
520:LRET ISZ 3,LOOP+2
522: LDM 8 /RESET Y
523: XCH 3
524: JMS DLY /DELAY FOR LIGHTS
526: JMS IXOP
528: BBL 0 /RETURN
529:/ ROM PORT 0 OUTPUT ROUTINE
529:/
529:RPOOP FIM OP,0 /SELECT ROM 0
531: SRC OP
532: WRR /OUTPUT ACC TO PORT
533: FIM OP,16 /SELECT ROM 1
535: SRC OP
536: BBL 0 /RETURN
537:/ DELAY ROUTINE
537:/
537:DLY FIM OP,250 /DELAY 1/2 MS
539: ISZ 0,*
541: ISZ 1,*
543: BBL 0
544:/ SCORE MATCH ROUTINE
544:/
544:MATCH CLC /GET A TENS DIGIT MATCH
545: LD 11
546: ADD 12
547: ADD 13
548: CLC
549: DAA
550: XCH 0
551: BBL 0
552:/ SET BIT ROUTINES
552:/
552:SBI RAR /SET BIT 1
553: RAR
554: STC
555: RAL
556: RAL
557: XCH 0
558: BBL 0
559:SB2 RAL /SET BIT 2
560: RAL
561: STC
562: RAR
563: RAR
564: XCH 0
565: BBL 0
566:/ SOLENOID OUTPUT ROUTINE
566:SOLEN LDM 4 /ISSUE DCL 3-0
567: DCL
568:WRR CLB

```

244

0: =258
 258:TABLE =488
 258:LRET =520
 258:SOLE =566
 258:SB1 =552
 258:SB2 =559
 258:GMINC =250
 258:KCH =245
 258:IXOP =238
 258:/SWITCH SERVICE VECTOR ROUTINE
 258:/
~~258:VECTR INC 4 /INCREMENT HIT COUNT~~
 259: LD 5 /TEST MISS COUNT FOR NEW SWITCH
 260: CLC
 261: RAR
~~262: JAZ LRET /SAME OLD SWITCH~~
 264: FIM OP, TABLE /GET JUMP TABLE START ADDRESS
 10A 266: LD 2 /ADD LEAST SIGNIFICANT PART OF OFFSET
 267: RAR
 10C 268: LD 3A
 269: JCZ **5
 271: RAL
 272: CLC
 273: RAR
 274: CLC
 275: ADD 1
 276: XCH 1
 277: JCZ **3
 279: INC 0
 280: LD 2 /ADD MOST SIGNIFICANT PART OF OFFSET
 281: RAR
 282: JCZ **6
 284: RAR
 285: JCZ **3
 287: INC 0
 288: JIN OP /JUMP TO SERVICE ROUTINE
 289:/SWITCH SERVICE ROUTINES
 289:/
 289:SSHOT JMS SOLEN /SLING SHOTS
 291: JUN SCT
 293:TBUMP JMS SOLEN /THUMPER BUMPERS
 295: LD 3 /TEST IF GREEN OR YELLOW
 296: RAR
~~297: JCO SCT~~
 299: LD 9 /TEST POINT VALUE OF GTB
 300: RAL
 301: JCO SCH
 303: JUN SCT
 305:E LD 7 /E ROLL OVER
 306: RAR
 307: STC
 308: RAL
 309: XCH 7
 310: JUN SCH
 312:L LD 7 /L ROLL OVER
 313: JMS SB1
 315: LD 0
 316: JUN E+4
 318:T LD 7 /T ROLL OVER
 319: JMS SB2
 321: LD 0
 322: JMS E+4
~~323: LD 6 /T ROLL OVER~~

VECTR,

LD 8
 RAL
 JCZ **4
 JUN LRET
 RAL
 RAL
 JCZ **4
 JUN LRET
 INC 4 /INCREMENT HIT COUNT

/TEST FOR SEAS TILT

JAN **4
 JUN LRET

FIN OP

JCZ SCH

4.1, LD 7
 RAL
 STC
 RAR
 XCH 7

327: LD 0

328: XCH 6

329: LD 9 /SET GTB

330: RAL

331: STC

332: RAR

333: XCH 9

334: JUN SCK

336: R LD 6 /R ROLL OVER

337: RAR

338: STC

339: RAL

340: JUN 01+4

342: 02 LD 7 /02 ROLL OVER

343: RAL

344: STC

345: RAR

346: JUN E+4

348: MR1 LD 9 /MUSHROOM 1

349: RAR

350: STC

351: RAL

352: XCH 9

353: JUN SCK

355: MR2 LD 9 /MUSHROOM 2

356: JMS SB1

358: LD 0

359: XCH 9

360: JUN SCH

362: MR3 LD 9 /MUSHROOM 3

363: JMS SB2

365: LD 0

366: JUN MR1+4

368: RA1 LD 9 /RIGHT ALLEY 1

369: RAR

370: JCO SCK

372: JUN SCH

374: RA2 LD 9 /RIGHT ALLEY 2

375: RAR

376: JUN RA1+1

378: RA3 LD 9 /RIGHT ALLEY 3

379: RAR

380: JUN RA2+1

382: RA4 LD 9 /RIGHT ALLEY RESET

383: RAL

384: LDM 0

385: RAR

386: XCH 9

387: JUN LRET

389: XTRA LD 6 /EXTRA ROLL OVER

390: RAL

391: RAL

392: JCZ SCK

394: LD 8

395: RAR

396: STC

397: RAL

398: XCH 8

399: JUN SCK

401: SPEC LD 6 /SPECIAL ROLL OVER

402: RAL

403: JCZ SCK

405: JMS GMINC

407: JUN SCK

409: / SCORE ROUTINES

XCH 6

JUN 01+6

02, LD 6

JMS SB1

LD 0

XCH 6

JUN SCH

409: SCT LDM 0 /OUTPUT 10 PT CHIME
 410: JMS KCH
 412: LD 7 /TEST FOR EXTRA & SPECIAL
 413: CMA
 414: JAN **24
 416: LD 6
 417: RAR
 418: JCZ **20
 420: RAR
 421: JCZ **17
 423: LD 10 /TEST EVEN ODD TENS DIGIT
 424: RAR
 425: JCZ **11
 427: LD 8 /TEST SHOOT AGAIN
 428: RAR
 429: LDM 3
 430: JCO **3
 432: LDM 7
 433: XCH 6
 434: JUN **4
 436: LDM 11
 437: XCH 6
 438: LD 10 /INCREMENT SCORE
 439: IAC
 440: DAA
 441: XCH 10
 442: JCO SCH+3
 444: JUN LRET
 446: /
 446: SCH LDM 1 /OUTPUT 100 PT CHIME
 447: JMS KCH
 449: LD 11 /INCREMENT SCORE
 450: IAC
 451: DAA
 452: XCH 11
 453: JCO SCH+3
 455: JUN LRET
 457: /
 457: SCK LDM 2 /OUTPUT 1,000 PT CHIME
 458: JMS KCH
 460: LD 12 /INCREMENT SCORE
 461: IAC
 462: DAA
 463: XCH 12
 464: JCZ **20
~~466: INC 13 /INCREMENT 10,000 DIGIT TEST FREE PLAY~~
 467: LD 13
 468: DAC
 469: DAC
 470: VRR
 471: RAL
 472: JCO **12
 474: LDM 5
 475: JMS IXOP+2 /LOOK AT FREE PLAY SWITCHES
 477: JTZ **4
 479: JMS GMING /FREE PLAY
 481: JMS IXOP+1 /RETURN X TO WHERE IT WAS
 483: JUN LRET
 485: =488
 488: O+SSHOT /Y=0-7 X=B
 489: O+SSHOT
 490: O+TBUMP
 491: O+TBUMP
 492: O+TBUMP
 493: O+TBUMP

JCZ **26

LD 13 /INCREMENT 10,000
 IAC
 DAA
 XCH 13
 DIGIT-TEST FREE
 PLAY

XCH 0
 LD 0

JCO **14

LD 0
 WRR

494: 0+L
495: 0+T
496: 0+02 /Y=0-7,X=C
497: 0+R
498: 0+01
499: 0+MR1
500: 0+MR2
501: 0+MR3
502: 0+RA1
503: 0+RA2
504: 0+RA3 /Y=0-6,X=D
505: 0+RA4
506: 0+SCT
507: 0+SCK
508: 0+XTRA
509: 0+SPEC
510: 0+SCK

loc 570 25A U/P: 0 TS PBL 0 Marine Lnc CO

P/2 106 JAZ JAZ 14
263 107 LMT BC BC

0 0 LDM 12 JUN 42
1 1 XCH 8 SBO 44

SBO 245 LDM 12 DC
1 245 XCH 8 BC
2 246 LDM 15 DF
3 247 XCH 5 BS
4 248 FIM OP 20
5 249 16 10
6 24A SPC OP 21
7 24B JUN 40
SBO 24C 2 02

2 LDM 8 DF
3 3 JMS SO
4 4 IXPP +2 FO
5 5 WRR EZ
6 6 JTE 11
7 7 06
8 8 LD 14 AE
9 9 JAN IC
10 10 * +4 OD
11 11 JUN 40
12 12 17 11
13 13 LDM 13 14 DE
14 14 JUN 42
15 15 S89 40

S89 24D JMS 52
90 E RPOPP 11
1 F LDM 5 25
2 250 SMS 50
3 1 IXPP +2 FO
4 3 LDM 6 5
5 4 JMS 5
6 5 IXOP +2 FO
7 6 JUN 40
S89 25L 17 11

Exhibit # 2
J.D. 55
1-11-78
J.A.P., Notary

Loc Dec	Loc Hex	NAME INIC	NEW Hex	OLD Hex
2	2	LDM 0	D0	
3	3	JMS	52	
4	4	RP00P	11	
5	5	LDM 0	D0	
6	6	JMS =	50	
7	7	IX0P+2	F0	
8	8	WRR	E2	
9	9	JT2	11	
10	A	*	09	
11	B	LD 14	AE	
12	C	JAN	LC	
13	D	*+8	14	
14	E	LDM 8	D8	
15	F	XCH 0	B0	
16	10	ISZ 0,	70	
17	1	*	10	
18	2	JUN	40	
19	3	*+8	1A	
20	4	LDM 14	DE	
21	5	JME	52	
22	6	RP00P	11	
23	7	LDM 0	D0	
24	8	JMS	52	
25	9	RP00P	11	
26	A	LDM 1	D1	
27	B	WRR	E2	
28	C	CLB	F0	
29	D	JT2	11	
30	E	*+3	20	
31	F	IAC	F2	
32	20	XCH 4	B4	
33	11	FIM 1P	22	
34	12	18	08	
35	13	LDM 18 15	DJ	
36	14	XCH 0	B0	
37	15	ISZ 0,	70	
38	16	*	25	
39	17	LD 4	A4 ✓	
40	18	LAR	F6 ✓	
41	19	LD 7	A7	
42	A	WOP	D0	
43	B	NDP	D0	
44	C	JUN	42	
45	2D	589	4D	

Loc	Sec	Loc	Hex	NAMEWORK	New Hex	Old Hex
589		24D		JCP	12	
90		E		*+3	50	
1		F		LD 9	A9	
2		25D		WRR	E2	
3		1		LDM 15	DC DF	
4		2		JMS	52	
5		3		RPOOP	11	
6		4		LDM 0	D0	
7		5		JMS	52	
8		6		RPOOP	11	
9		7		LD 4	A4	
600		8		RAR	F6	
1		9		LD 6	A6	
2		A		JCP	12	
3		B		*+3	5D	
4		C		LD 8	A8	
5		D		WRR	E2	
6		E		LDM 13	DD	
7		F		JMS	52	
8		260		RPOOP	11	
9		1		LDM 0	D0	
610		2		JMS	52	
1		3		RPOOP	11	
2		4		WRR	E2	
3		5		JT 8	19	
4		6		7	65	
5		7		JUN	40	
611		268		46	2E	
236		EC		JUN	42	
237		ED		17	69	
17		269		LDM 8	D8	
8		A		JMS	50	
9		B		IXP1+2	F0	
20		C		JUN	40	
21		265		P.WBAL	02	
224		EO		JUN	42	
225		E1		622	6E	
622		26E		LDM 8	D8	
3		F		JMS	50	
4		270		IXOP 42	F0	
5		1		JUN	40	
226		271		226	E2	

42
50

31
32

NOP
NOP 00
00

466
467

1D2
1D3

JUN 43
806 26

806
7
8
9
0
1
512

326
7
8
9
A
B
32C

LD 13 AD
IAC FZ
DAA FB
XCH 13 BD
LD 12 AD
JUN 41
468 34

73
74

49
4A

JUN 43
813 23

813
4
5
6
817

32D
E
F
330
331

DAC F8
XCH 14 BE
LD 14 AE
JUN 40
75 4B

475
476

1DB
1DC

JUN 43
818 32

818
9
20
1
2
3

332
3
4
5
6
337

JMS 50
IXP+3 50
LD 0 A0
WRR E2
JUN 41
477 DD

470
471

1D6
1D7

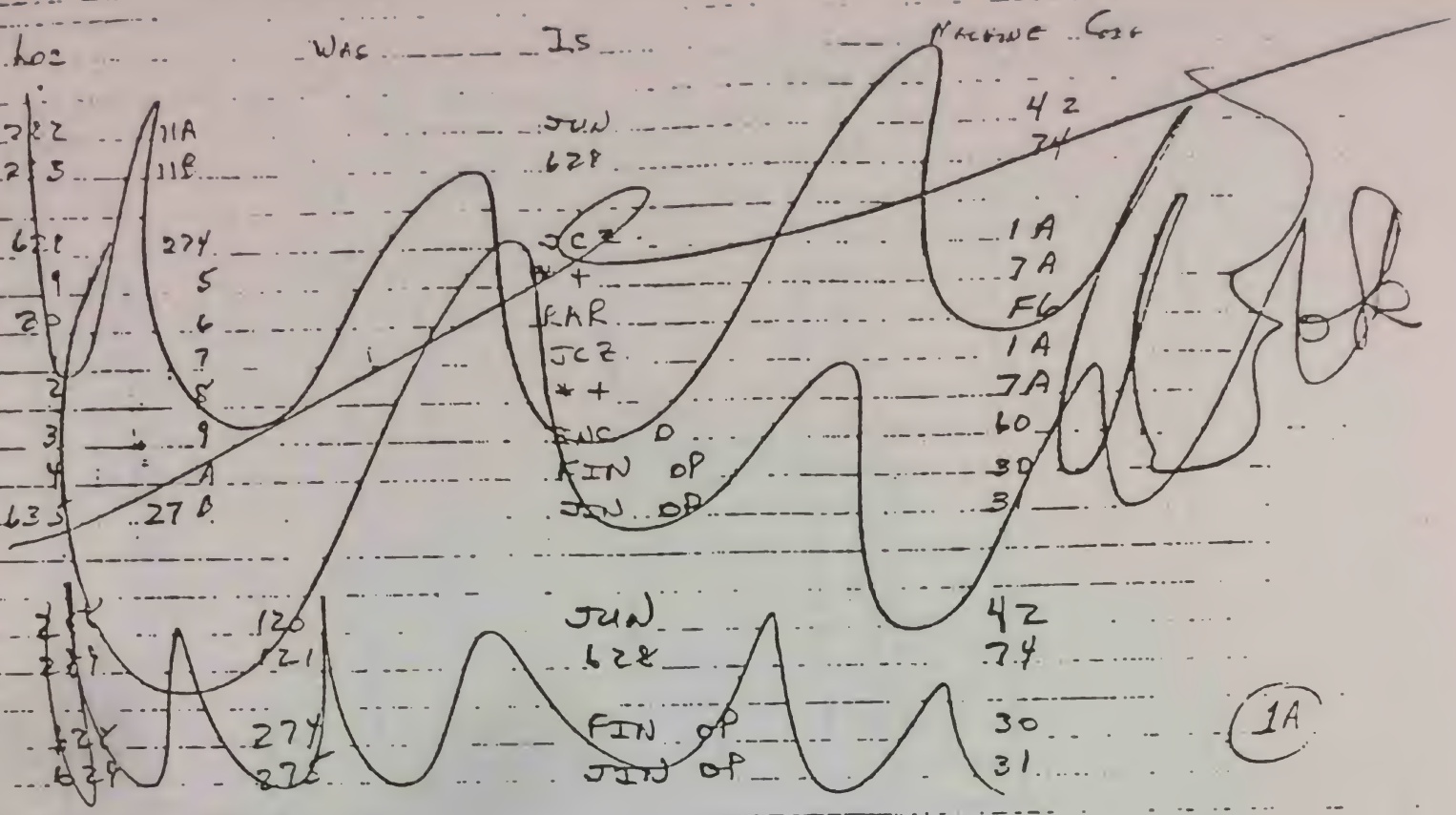
JUN 43
824 38

824
825
6
7
28

338
9
A
B
33C

XCH 0 B0
LD 0 A0
RAI F
JUN 41
472 08

Loc	Loc	Hex	NUMERONIC	NEW Hex	OLD Hex
19	BD	JUN	42		
190	BE	669	98		
664	291	JTC	19		
5	19	*	98		
6	A	JMS	50		
7	B	GMINC	FA		
8	C	JUN	40		
664	29D	191	BF		
192	CD	JUN	40		
193	CL	204	CC		
198	CL	JUN	43		
199	C7	672	00		
672	300	JTC	19		
8	301	*	00		
8	302	JMS	50		
6	303	GMINC	FA		
6	304	JUN	40		
677	305	200	CB		
202	CA	JMS	50		
203	CB	GMINC	FA		
227	E3	JAN	1C		
228	E4	447	EA		
125	4D	128	80		
21	51	*	50		40
43	D	LDN 14	3C		
591	24F	LDN 0	3C		
592	250	JUN	43		
593	251	678	06		
678	306	JMS	50		
9	7	IX01+2	FD		
80	P	LDN 0	DD		
1	9	JMS	52		
2	0	RPOPP	11		
3	E	JUN	42		
684	30C	594	52		



282	11A	JUN	42
283	11B	628	74
628	274	JCE	1A
9	5	27A	7A
30	6	RAR	FL
1	7	JCE	7A
2	1	27A	7A
3	9	INC D	60
4	A	JUN	41
5	275	284	1C
284	11C	FIN OP	30
285	11D	JUN OP	31

125 (boxed)

Loc	src	Loc	Hex	NAME/MONIC	New Hex	Old Hex
484		1E4		IRET	08	
					21	
488		1E8			21	
489		9			25	
490		A			25	
491		B			25	
492		C			31	
493		D			38	
494		E			3E	
495		F			56	
496		3F0			50	
497		1			44	
498		2			5C	
499		3			63	
500		4			6A	
501		5			70	
502		6			76	
503		1F7			7A	
504		8			7E	
505		9			99	
506		A			69	
507		B			85	
508		C			91	
509		D			C9	
510		1FE				
324		14F		LD 7	A7	
325		5		RAL	F5	
326		C		STC	FA	
327		7		RAR	F6	
328		148		XCH 7	B7	
342		150		LD 6	A6	
343		7		JMS	52	
344		8		SB1	28	
345		9		JUN	42	
346		15A		636	7C	
636		27C		LD 0	A0	
7		D		XCH 6	B6	
18		E		JUN	41	
639		27F		5CH	BE	
340		154		340	42	
341		155		630	80	

Loc Dec	Loc Hex	Nomenclature	New Hex	Old Hex
1640	280	XCH 6	B6	
✓ 642	281	JUN	41	
✓ 45	282	329	49	
✓ 3	1D1	*+19	E3	
✓ 46	1D9	*+11	E3	
47	2E	JUN	42	
	2F	643	83	
✓ 43	283	LDM 9	D9	
4	4	JMS	50	
5	5	IXP + 2	F0	
6	6	JUN	40	
647	287	48	30	
297	129	JCH	12	
298	12A	SCH	BE	
299	121	JMS L W	52	
5	21A	244	F4	
258	102	JUN	42	
259	103	648	88	
64P	288	LD 8	A8	
9	9	RAL	F5	
50	A	JCH	1E	
1	B	*+4	8E	
2	C	JUN	42	
3	D	LRET	08	
4	E	RAL	F5	
✓ 5	F	RAL	F5	
6	290	JCE	1A	
7	1	*+4	94	
8	2	JUN	42	
9	3	LRET	08	
60	4	INC 4	64	
11	5	LD 5	A5	
12	6	JUN	41	
13	297	260	04	

BASIC INSTRUCTION SET

The basic instruction set of the 4040 and 4004 (CPU) are shown below. The following section will describe each instruction in detail.

[Those instructions preceded by an asterisk (*) are 2 word instructions that occupy 2 successive locations in ROM]

MACHINE INSTRUCTIONS (Logic 1 = Low Voltage = Negative Voltage; Logic 0 = High Voltage = Ground)

MNEMONIC	OPR D ₃ D ₂ D ₁ D ₀	OPA D ₃ D ₂ D ₁ D ₀	DESCRIPTION OF OPERATION
NOP	0 0 0 0	0 0 0 0	No operation.
*JCN	0 0 0 1 A ₂ A ₂ A ₂ A ₂	C ₁ C ₂ C ₃ C ₄ A ₁ A ₁ A ₁ A ₁	Jump to ROM address A ₂ A ₂ A ₂ A ₂ , A ₁ A ₁ A ₁ A ₁ (within the same ROM that contains this JCN instruction) if condition C ₁ C ₂ C ₃ C ₄ ⁽¹⁾ is true, otherwise skip (go to the next instruction in sequence).
*FIM	0 0 1 0 D ₂ D ₂ D ₂ D ₂	R R R 0 D ₁ D ₁ D ₁ D ₁	Fetch immediate (direct) from ROM Data D ₂ , D ₁ to index register pair location RRR. ⁽²⁾
SRC	0 0 1 0	R R R 1	Send register control. Send the address (contents of index register pair RRR) to ROM and RAM at X ₂ and X ₃ time in the Instruction Cycle.
FIN	0 0 1 1	R R R 0	Fetch indirect from ROM. Send contents of index register pair location 0 out as an address. Data fetched is placed into register pair location RRR.
JIN	0 0 1 1	R R R 1	Jump indirect. Send contents of register pair RRR out as an address at A ₁ and A ₂ time in the Instruction Cycle.
*JUN	0 1 0 0 A ₂ A ₂ A ₂ A ₂	A ₃ A ₃ A ₃ A ₃ A ₁ A ₁ A ₁ A ₁	Jump unconditional to ROM address A ₃ , A ₂ , A ₁ .
*JMS	0 1 0 1 A ₂ A ₂ A ₂ A ₂	A ₃ A ₃ A ₃ A ₃ A ₁ A ₁ A ₁ A ₁	Jump to subroutine ROM address A ₃ , A ₂ , A ₁ , save old address. (Up 1 level in stack.)
INC	0 1 1 0	R R R R	Increment contents of register RRRR. ⁽³⁾
*ISZ	0 1 1 1 A ₂ A ₂ A ₂ A ₂	R R R R A ₁ A ₁ A ₁ A ₁	Increment contents of register RRRR. Go to ROM address A ₂ , A ₁ (within the same ROM that contains this ISZ instruction) if result ≠ 0, otherwise skip (go to the next instruction in sequence).
ADD	1 0 0 0	R R R R	Add contents of register RRRR to accumulator with carry.
SUB	1 0 0 1	R R R R	Subtract contents of register RRRR to accumulator with borrow.
LD	1 0 1 0	R R R R	Load contents of register RRRR to accumulator.
XCH	1 0 1 1	R R R R	Exchange contents of index register RRRR and accumulator.
BBL	1 1 0 0	D D D D	Branch back (down 1 level in stack) and load data DDDD to accumulator.
LDM	1 1 0 1	D D D D	Load data DDDD to accumulator.

NEW 4040 INSTRUCTIONS

MNEMONIC	OPR D ₃ D ₂ D ₁ D ₀	OPA D ₃ D ₂ D ₁ D ₀	DESCRIPTION OF OPERATION
HLT	0 0 0 0	0 0 0 1	Halt - Inhibit program counter and data buffers.
BBS	0 0 0 0	0 0 1 0	Branch Back from Interrupt and restore the previous SRC. The Program Counter and send register control are restored to their pre-interrupt value.
LCR	0 0 0 0	0 0 1 1	The contents of the COMMAND REGISTER are transferred to the ACCUMULATOR.
OR4	0 0 0 0	0 1 0 0	The 4 bit contents of register #4 are logically "OR-ed" with the ACCUM.
OR5	0 0 0 0	0 1 0 1	The 4 bit contents of index register #5 are logically "OR-ed" with the ACCUMULATOR.
AN6	0 0 0 0	0 1 1 0	The 4 bit contents of index register #6 are logically "AND-ed" with the ACCUMULATOR.
AN7	0 0 0 0	0 1 1 1	The 4 bit contents of index register #7 are logically "AND-ed" with the ACCUMULATOR.
DB0	0 0 0 0	1 0 0 0	DESIGNATE ROM BANK 0. CM-ROM ₀ becomes enabled.
DB1	0 0 0 0	1 0 0 1	DESIGNATE ROM BANK 1. CM-ROM ₁ becomes enabled.
SB0	0 0 0 0	1 0 1 0	SELECT INDEX REGISTER BANK 0. The index registers 0 - 7.
SB1	0 0 0 0	1 0 1 1	SELECT INDEX REGISTER BANK 1. The index registers 0* - 7*.
EIN	0 0 0 0	1 1 0 0	ENABLE INTERRUPT.
DIN	0 0 0 0	1 1 0 1	DISABLE INTERRUPT.
RPM	0 0 0 0	1 1 1 0	READ PROGRAM MEMORY.

INPUT/OUTPUT AND RAM INSTRUCTIONS

(The RAM's and ROM's operated on in the I/O and RAM instructions have been previously selected by the last SRC instruction executed.)

MNEMONIC	OPR D ₃ D ₂ D ₁ D ₀	OPA D ₃ D ₂ D ₁ D ₀	DESCRIPTION OF OPERATION
WRM	1 1 1 0	0 0 0 0	Write the contents of the accumulator into the previously selected RAM main memory character.
WMP	1 1 1 0	0 0 0 1	Write the contents of the accumulator into the previously selected RAM output port.
WRR	1 1 1 0	0 0 1 0	Write the contents of the accumulator into the previously selected ROM output port. (I/O Lines)
WPM	1 1 1 0	0 0 1 1	Write the contents of the accumulator into the previously selected half byte of read/write program memory (for use with 4008/4009 only)
WR ₀ ⁽⁴⁾	1 1 1 0	0 1 0 0	Write the contents of the accumulator into the previously selected RAM status character 0.
WR ₁ ⁽⁴⁾	1 1 1 0	0 1 0 1	Write the contents of the accumulator into the previously selected RAM status character 1.
WR ₂ ⁽⁴⁾	1 1 1 0	0 1 1 0	Write the contents of the accumulator into the previously selected RAM status character 2.
WR ₃ ⁽⁴⁾	1 1 1 0	0 1 1 1	Write the contents of the accumulator into the previously selected RAM status character 3.
SBM	1 1 1 0	1 0 0 0	Subtract the previously selected RAM main memory character from accumulator with borrow.
RDM	1 1 1 0	1 0 0 1	Read the previously selected RAM main memory character into the accumulator.
RDR	1 1 1 0	1 0 1 0	Read the contents of the previously selected ROM input port into the accumulator. (I/O Lines)
ADM	1 1 1 0	1 0 1 1	Add the previously selected RAM main memory character to accumulator with carry.
RD ₀ ⁽⁴⁾	1 1 1 0	1 1 0 0	Read the previously selected RAM status character 0 into accumulator.
RD ₁ ⁽⁴⁾	1 1 1 0	1 1 0 1	Read the previously selected RAM status character 1 into accumulator.
RD ₂ ⁽⁴⁾	1 1 1 0	1 1 1 0	Read the previously selected RAM status character 2 into accumulator.
RD ₃ ⁽⁴⁾	1 1 1 0	1 1 1 1	Read the previously selected RAM status character 3 into accumulator.

ACCUMULATOR GROUP INSTRUCTIONS

CLB	1 1 1 1	0 0 0 0	Clear both. (Accumulator and carry)
CLC	1 1 1 1	0 0 0 1	Clear carry.
IAC	1 1 1 1	0 0 1 0	Increment accumulator.
CMC	1 1 1 1	0 0 1 1	Complement carry.
CMA	1 1 1 1	0 1 0 0	Complement accumulator.
RAL	1 1 1 1	0 1 0 1	Rotate left. (Accumulator and carry)
RAR	1 1 1 1	0 1 1 0	Rotate right. (Accumulator and carry)
TCC	1 1 1 1	0 1 1 1	Transmit carry to accumulator and clear carry.
DAC	1 1 1 1	1 0 0 0	Decrement accumulator.
TCS	1 1 1 1	1 0 0 1	Transfer carry subtract and clear carry.
STC	1 1 1 1	1 0 1 0	Set carry.
DAA	1 1 1 1	1 0 1 1	Decimal adjust accumulator.
KBP	1 1 1 1	1 1 0 0	Keyboard process. Converts the contents of the accumulator from a one out of four code to a binary code.
DCL	1 1 1 1	1 1 0 1	Designate command line.

NOTES: (1) The condition code is assigned as follows:

C₁ = 1 Invert jump condition C₂ = 1 Jump if accumulator is zero C₄ = 1 Jump if test signal is a 0
C₁ = 0 Not Invert jump condition C₃ = 1 Jump if carry/link is a 1

(2) RRR is the address of 1 of 8 index register pairs in the CPU.

(3) RRRR is the address of 1 of 16 index registers in the CPU.

(4) Each RAM chip has 4 registers, each with twenty 4-bit characters subdivided into 16 main memory characters and 4 status characters. Chip number, RAM register and main memory character are addressed by an SRC instruction. For the selected chip and register, however, status character locations are selected by the instruction code (OPA).

DETAILED INSTRUCTION DESCRIPTION

A. Symbols and Abbreviations

The following symbols and abbreviations will be used throughout the next few sections:

SRCR	SRC Register
()	the content of is transferred to
ACC	Accumulator (4 bit)
CY	Carry Flip-Flop
ACBR	Accumulator Buffer Register (4 bit)
RRRR	Index register address
RRR	Index register pair address
P _L	Low order program counter Field (4 bit)
P _M	Middle order program counter Field (4 bit)
P _H	High order program counter Field (4 bit)
a _i	Order i content of the accumulator
CM _i	Order i content of the command register
M	RAM main character location
M _i	RAM status character i
DB (T)	Data bus content at time T
Stack	The 3 or 7 registers in the address register other than the program counter.
CR	Command register
IE	Interrupt enable
RB0	Register bank 0 RRRR ₀ — RRRR, enable
RB1	Register bank 1 RRRR ₀ — RRRR, enable
V	Logical OR
Λ	Logical AND

Throughout the text "page" means a block of 256 instructions whose address differs only on the most significant 4 bits.

Example: page 7 means all locations having addresses between 0111 0000 0000 and 0111 1111 1111

B. Format for Describing Each Instruction

Each instruction will be described as follows:

- (1) Mnemonic symbol and meaning
- (2) OPR and OPA code
- (3) Symbolic representation of the instruction
- (4) Description of the instruction (if necessary)
- (5) Example and/or exceptions (if necessary)

C. One Word Machine Instructions

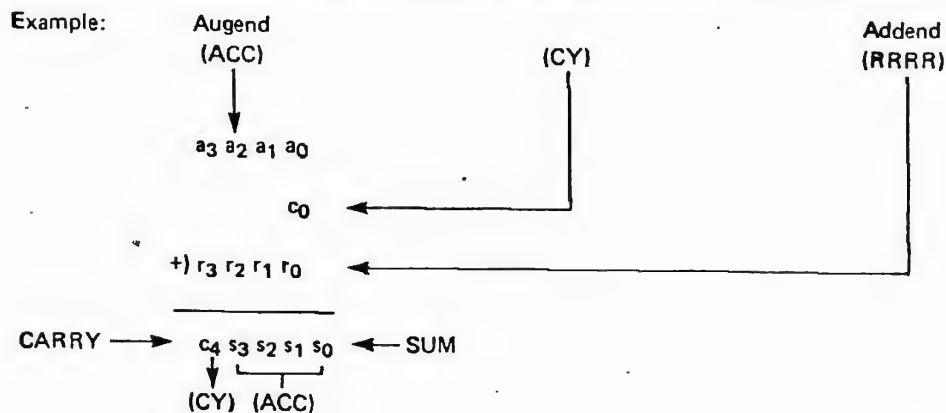
Mnemonic:	NOP (No Operation)
OPR OPA:	0000 0000
Symbolic:	Not applicable
Description:	No operation performed

Mnemonic:	LDM (Load Data to Accumulator)
OPR OPA:	1101 DDDD
Symbolic:	DDDD → ACC
Description:	The 4 bits of data, DDDD stored in the OPA field of instruction word are loaded into the accumulator. The previous contents of the accumulator are lost. The carry/link bit is unaffected.

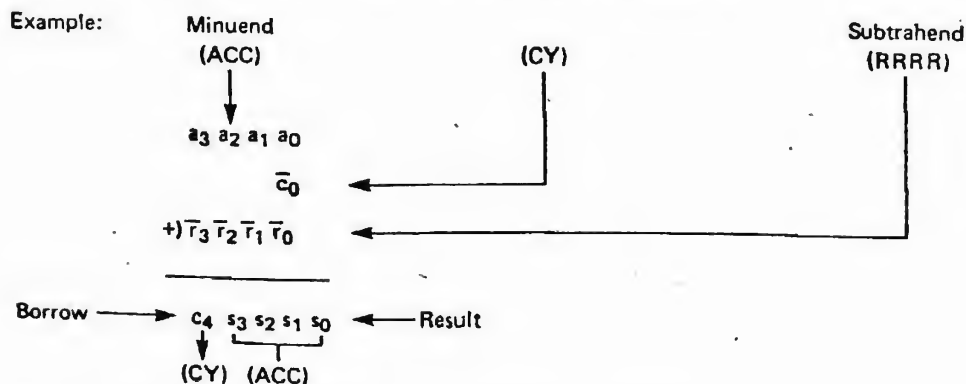
Mnemonic: LD (Load index register to Accumulator)
OPR OPA: 1010 RRRR
Symbolic: (RRRR) → ACC
Description: The 4 bit content of the designated index register (RRRR) is loaded into the accumulator. The previous contents of the accumulator are lost. The 4 bit content of the index register and the carry/link bit are unaffected.

Mnemonic: XCH (Exchange index register and accumulator)
OPR OPA: 1011 RRRR
Symbolic: (ACC) → ACBR, (RRRR) → ACC, (ACBR) → RRRR
Description: The 4 bit content of the designated index register is loaded into the accumulator. The prior content of the accumulator is loaded into the designated register. The carry/link bit is unaffected.

Mnemonic: ADD (Add index register to accumulator with carry)
OPR OPA: 1000 RRRR
Symbolic: (RRRR) + (ACC) + (CY) → ACC, CY
Description: The 4 bit content of the designated index register is added to the content of the accumulator with carry. The result is stored in the accumulator. The carry/link is set to 1 if a sum greater than 15_{10} was generated to indicate a carry out; otherwise, the carry/link is set to 0. The 4 bit content of the index register is unaffected.



Mnemonic: SUB (Subtract index register from accumulator with borrow)
OPR OPA: 1001 RRRR
Symbolic: (ACC) + (RRRR) + (CY) → ACC, CY
Description: The 4 bit content of the designated index register is complemented (ones complement) and added to content of the accumulator with borrow and the result is stored in the accumulator. If a borrow is generated, the carry bit is set to 0; otherwise, it is set to 1. The 4 bit content of the index register is unaffected.



Mnemonic: INC (Increment index register)
OPR OPA: 0110 RRRR
Symbolic: (RRRR) + 1 → RRRR
Description: The 4 bit content of the designated index register is incremented by 1. The index register is set to zero in case of overflow. The carry/link is unaffected.

Mnemonic: BBL (Branch back and load data to the accumulator)
OPR OPA: 1100 DDDD
Symbolic: (Stack) → P_L, P_M, P_H; DDDD → ACC
Description: The program counter (address stack) is pushed down one level. Program control transfers to the next instruction following the last jump to subroutine (JMS) instruction. The 4 bits of data DDDD stored in the OPA portion of the instruction are loaded to the accumulator. BBL is used to return from subroutine to main program.

Mnemonic: JIN (Jump indirect)
OPR OPA: 0011 RRR1
Symbolic: (RRR0) → P_M
 (RRR1) → P_L; P_H unchanged
Description: The 8 bit content of the designated index register pair is loaded into the low order 8 positions of the program counter. Program control is transferred to the instruction at that address on the same page (same ROM) where the JIN instruction is located. The 8 bit content of the index register is unaffected.

EXCEPTIONS: When JIN is located at the address (P_H) 1111 1111 program control is transferred to the next page in sequence and not to the same page where the JIN instruction is located. That is, the next address is (P_H + 1) (RRR0) (RRR1) and not (P_H) (RRR0) (RRR1)

Mnemonic: SRC (Send register control)
OPR OPA: 0010 RRR1
Symbolic: (RRR0) → DB (X₂)
 (RRR1) → DB (X₃)
Description: The 8 bit content of the designated index register pair is sent to the RAM address register at X₂ and X₃. A subsequent read, write, or I/O operation of the RAM will utilize this address. Specifically, the first 2 bits of the address designate a RAM chip; the second 2 bits designate 1 out of 4 registers within the chip; the last 4 bits designate 1 out of 16 4 bit main memory characters within the register. This command is also used to designate a ROM for a subsequent ROM I/O port operation. The first 4 bits designate the ROM chip number to be selected. The address in ROM or RAM is not cleared until the next SRC instruction is executed. The 8 bit content of the index register is unaffected.

Mnemonic: FIN (Fetch indirect from ROM)
OPR OPA: 0011 RRR0
Symbolic: (P_H) (0000) (0001) → ROM address
 (OPR) → RRR0
 (OPA) → RRR1
Description: The 8 bit content of the 0 index register pair (0000) (0001) is sent out as an address in the same page where the FIN instruction is located. The 8 bit word at that location is loaded into the designated index register pair. The program counter is unaffected; after FIN has been executed the next instruction in sequence will be addressed. The content of the 0 index register pair is unaltered unless index register 0 was designated.

EXCEPTIONS: a. Although FIN is a 1-word instruction, its execution requires two memory cycles (21.6 μsec).
 b. When FIN is located at address (P_H) 1111 1111 data will be fetched from the next page (ROM) in sequence and not from the same page (ROM) where the FIN instruction is located. That is, next address is (P_H + 1) (0000) (0001) and not (P_H) (0000) (0001).

Mnemonic: HLT
 OPR OPA: 0000 0001
 Symbolic: 1 → HALT 1 → STOP
 Description: The processor sets the HALT and STOP flip-flops. Program counter incrementer and data input buffers are inhibited. The processor executes NOP continuously; continuation can occur by means of STOP or INTERRUPT control.
 In this mode, the Program Counter + 1 is gated out at A₁, A₂, and A₃, times on the data bus. M₁, M₂ times will contain the addressed ROM instruction on the data bus. X₁, the 4 bit Accumulator contents, X₂ and X₃ will contain the 8 bit SRC register.

Mnemonic: BBS
 OPR OPA: 0000 0010
 Symbolic: (Stack → P_L, P_M, P_H;) SRCR0 → DB(X₂) SRCR1 → DB(X₃)
 Description: This instruction is a combination of BRANCH BACK and SRC. The effective address counter is decremented and program control is returned to the location saved by the forced JMS which occurred at the beginning of the Interrupt routine. In addition, the content of the SRC register is sent out at X₂ and X₃ of the instruction cycle, thus restoring the I/O port selection. This instruction will also turn off the INTA line re-enabling the CPU for Interrupt.
 The previously selected Index register bank will also be restored during this instruction.

Mnemonic: LCR
 OPR OPA: 0000 0011
 Symbolic: (CR) → ACC
 Description: The 4 bit contents of the COMMAND REGISTER are transferred to the ACCUMULATOR. This allows saving the command register values before processing the interrupt.

Mnemonic: OR4 -
 OPR OPA: 0000 0100
 Symbolic: (RRRR₄) V (ACC) → ACC
 Description: The 4 bit contents of index register #4 are logically "OR-ed" with the ACCUMULATOR. The result is placed in the ACCUMULATOR and the CARRY flip-flop is unaffected.

Examples: (ACC) 0101
 (RRRR₄) 1001
 ACC 1101

(ACC) 0000
 (RRRR₄) 1000
 ACC 1000

Mnemonic: OR5
 OPR OPA: 0000 0101
 Symbolic: (RRRR₅) V (ACC) → ACC
 Description: The 4 bit contents of index register #5 are logically "OR-ed" with the ACCUMULATOR. Carry flip-flop is unaffected.

Mnemonic: AN6
 OPR OPA: 0000 0110
 Symbolic: $(RRRR_6) \wedge (ACC) \rightarrow ACC$
 Description: The 4 bit contents of index register #6 are logically "AND-ed" with the ACCUMULATOR. The result is placed in the ACCUMULATOR and the CARRY is unaffected.

Examples:

(ACC)	0110
(RRRR ₆)	<u>0100</u>
ACC	0100

(ACC)	1111
(RRRR ₆)	<u>0001</u>
ACC	0001

Mnemonic: AN7
 OPR OPA: 0000 0111
 Symbolic: $(RRRR_7) \wedge (ACC) \rightarrow ACC$
 Description: The 4 bit contents of index register #7 are logically "AND-ed" with the ACCUMULATOR. Carry flip-flop is unaffected.

Mnemonic: DB0
 OPR OPA: 0000 1000
 Symbolic: Enable \rightarrow CM-ROM₀
 Description: DESIGNATE ROM BANK 0. The most significant bit of the COMMAND REGISTER, CR₃, is reset. On the third instruction cycle following its execution, it causes CM-ROM₀ to be activated. This Bank is selected with reset.

Mnemonic: DB1
 OPR OPA: 0000 1001
 Symbolic: Enable \rightarrow CM-ROM₁
 Description: DESIGNATE ROM BANK 1. The most significant bit of the COMMAND REGISTER, CR₃, is set. On the third instruction cycle following its execution, it causes CM-ROM₁ to be activated.

Mnemonic: SB0
 OPR OPA: 0000 1010
 Symbolic: 1 \rightarrow RB0, 0 \rightarrow RB1
 Description: SELECT INDEX REGISTER BANK 0. The index register bank select flip-flop is reset. Index registers 0 - 7, 8 - 15 will be available for program use. This bank is to be selected with a Reset.

Mnemonic: SB1
 OPR OPA: 0000 1011
 Symbolic: 0 \rightarrow RB0 1 \rightarrow RB1
 Description: SELECT INDEX REGISTER BANK 1. The index register bank select flip-flop is set. Index registers 0* - 7*, 8 - 15 will be available for program use.

Mnemonic: RPM
 OPR OPA: 0000 1110
 Symbolic: (1111) (SRC) \rightarrow ROM/RAM address
 (DDDD) \rightarrow ACC
 Description: READ PROGRAM MEMORY. This instruction can be used only with the 4289 Standard Memory and I/O Interface Chip. The contents of the previously selected nibble of R/W Program Memory are transferred to the 4040 and loaded to the ACCUMULATOR.

Mnemonic: EIN
 OPR OPA: 0000 1100
 Symbolic: 1 → IE
 Description: ENABLE INTERRUPT. Internal interrupt detection logic is enabled.

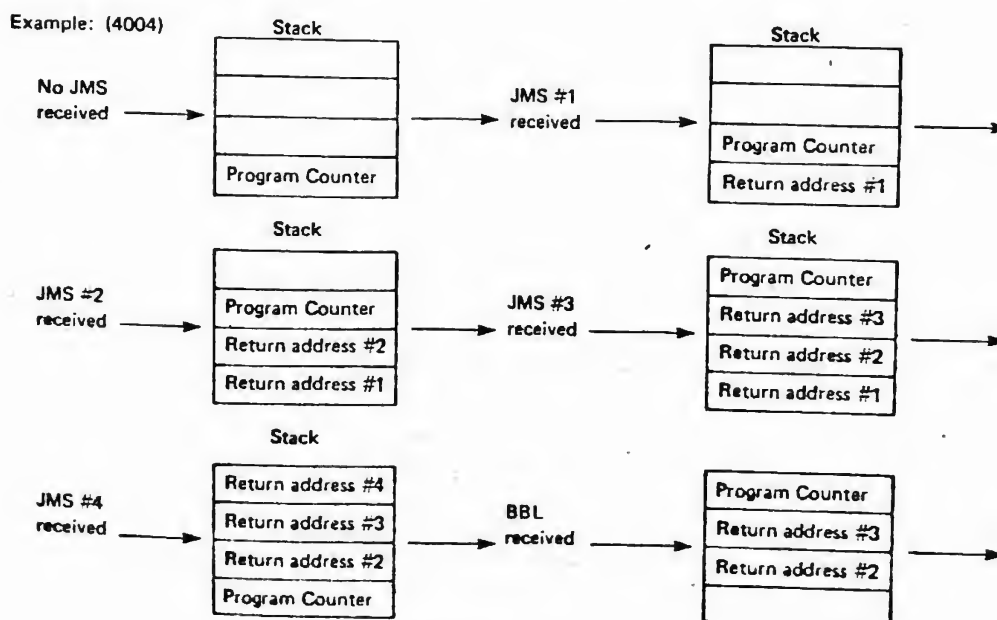
Mnemonic: DIN
 OPR OPA: 0000 1101
 Symbolic: 0 IE
 Description: DISABLE INTERRUPT. Internal interrupt detection logic is disabled.

D. Two Word Machine Instruction

Mnemonic: JUN (Jump unconditional)
 1st word OPR OPA: 0100 A₃ A₃ A₃ A₃
 2nd word OPR OPA: A₂ A₂ A₂ A₂ A₁ A₁ A₁ A₁
 Symbolic: A₁ A₁ A₁ A₁ → P_L, A₂ A₂ A₂ A₂ → P_M, A₃ A₃ A₃ A₃ → P_H
 Description: Program control is unconditionally transferred to the instruction locator at the address A₃ A₃ A₃ A₃, A₂ A₂ A₂ A₂, A₁ A₁ A₁ A₁

Mnemonic: JMS (Jump to Subroutine)
 1st word OPR OPA: 0101 A₃ A₃ A₃ A₃
 2nd word OPR OPA: A₂ A₂ A₂ A₂ A₁ A₁ A₁ A₁
 Symbolic: (P_H, P_M, P_L + 2) → Stack
 A₁ A₁ A₁ A₁ → P_L, A₂ A₂ A₂ A₂ → P_M, A₃ A₃ A₃ A₃ → P_H
 Description: The address of the next instruction in sequence following JMS (return address) is saved in the push down stack. Program control is transferred to the instruction located at the 12 bit address (A₃A₃A₃A₃A₂A₂A₂A₂A₁A₁A₁A₁). Execution of a return instruction (BBL) will cause the saved address to be pulled out of the stack, therefore, program control is transferred to the next sequential instruction after the last JMS.

The push down stack has 4 registers (8 registers in 4040). One of them is used as the program counter, therefore nesting of JMS can occur up to 3 levels (7 levels in the 4040).



The deepest return address is lost.

Mnemonic: JCN (Jump conditional)
1st word OPR OPA: 0001 C₁ C₂ C₃ C₄
2nd word OPR OPA: A₂ A₂ A₂ A₂ A₁ A₁ A₁ A₁
Symbolic: If C₁ C₂ C₃ C₄ is true, A₂ A₂ A₂ A₂ → P_M
A₁ A₁ A₁ A₁ → P_L, P_H unchanged
if C₁ C₂ C₃ C₄ is false,
(P_H) → P_H, (P_M) → P_M, (P_L + 2) → P_L
Description: If the designated condition code is true, program control is transferred to the instruction located at the 8 bit address A₂ A₂ A₂ A₂, A₁ A₁ A₁ A₁ on the same page (ROM) where JCN is located.
If the condition is not true the next instruction in sequence after JCN is executed.
The condition bits are assigned as follows:
C₁ = 0 Do not invert jump condition
C₁ = 1 Invert jump condition
C₂ = 1 Jump if the accumulator content is zero
C₃ = 1 Jump if the carry/link content is 1
C₄ = 1 Jump if test signal (pin 10 on 4004) is zero.

C_X Condition Table for JCN Instruction

C ₁	C ₂	C ₃	C ₄	
0	0	0	0	NO OPERATION
0	0	0	1	Jump if test = 0 (High)
0	0	1	0	Jump if CY = 1
0	0	1	1	Jump if test = 0 or CY = 1
0	1	0	0	Jump if AC = 0
0	1	0	1	Jump if test = 0 or AC = 0
0	1	1	0	Jump if CY = 1 or AC = 0
0	1	1	1	Jump if test = 0 or CY = 1 or AC = 0
1	0	0	0	Jump Unconditionally
1	0	0	1	Jump if test = 1 (Low)
1	0	1	0	Jump if CY = 0
1	0	1	1	Jump if test = 1 and CY = 0
1	1	0	0	Jump if AC ≠ 0
1	1	0	1	Jump if test = 1 and AC ≠ 0
1	1	1	0	Jump if CY = 0 and AC ≠ 0
1	1	1	1	Jump if test = 1 and CY = 0 and AC ≠ 0

Example: OPR OPA
0001 0110 Jump if accumulator is zero or carry = 1

Several conditions can be tested simultaneously.

The logic equation describing the condition for a jump is given below:

$$JUMP = \overline{C_1} \cdot ((ACC = 0) \cdot C_2 + (CY = 1) \cdot C_3 + \overline{TEST} \cdot C_4) +$$

$$C_1 \cdot ((ACC = 0) \cdot C_2 + (CY = 1) \cdot C_3 + TEST \cdot C_4)$$

EXCEPTIONS:

If JCN is located on words 254 and 255 of a ROM page, when JCN is executed and the condition is true, program control is transferred to the 8 bit address on the next page where JCN is located.

Mnemonic: ISZ (Increment index register skip if zero)
1st word OPR OPA: 0111 RRRR
2nd word OPR OPA: $A_2 A_2 A_2 A_2 A_1 A_1 A_1 A_1$
Symbolic: $(RRRR) + 1 \rightarrow RRRR$, if result = 0
 $(P_H) \rightarrow P_H$, $(P_M) \rightarrow P_M$, $(P_L + 2) \rightarrow P_L$;
if result $\neq 0$ $(P_H) \rightarrow P_H$,
 $A_2 A_2 A_2 A_2 \rightarrow P_M$, $A_1 A_1 A_1 A_1 \rightarrow P_L$
Description: The content of the designated index register is incremented by 1. The accumulator and carry/link are unaffected. If the result is zero, the next instruction after ISZ is executed. If the result is different from 0, program control is transferred to the instruction located at the 8 bit address $A_2 A_2 A_2 A_2 A_1 A_1 A_1 A_1$ on the same page (ROM) where the ISZ instruction is located.
EXCEPTIONS: If ISZ is located on words 254 and 255 of a ROM page, when ISZ is executed and the result is not zero, program control is transferred to the 8 bit address located on the next page in sequence and not on the same page where ISZ is located.

Mnemonic: FIM (Fetched immediate from ROM)
1st word OPR OPA: 0010 RRR0
2nd word OPR OPA: $D_2 D_2 D_2 D_2 D_1 D_1 D_1 D_1$
Symbolic: $D_2 D_2 D_2 D_2 \rightarrow RRR0$
 $D_1 D_1 D_1 D_1 \rightarrow RRR1$
Description: The 2nd word represents 8 bits of data which are loaded into the designated index register pair.

E. Input/Output Instructions

The following I/O instructions are described as they relate to ROM and RAM devices. These same instructions (mnemonics) can be redefined for devices other than ROM and RAM.

Mnemonic: RDM (Read RAM character)
OPR OPA: 1110 1001
Symbolic: $(M) \rightarrow ACC$
Description: The content of the previously selected RAM main memory character is transferred to the accumulator. The carry/link is unaffected. The 4 bit data in memory is unaffected.

Mnemonic: RDO (Read RAM status character 0)
OPR OPA: 1110 1100
Symbolic: $(M_{S0}) \rightarrow ACC$
Description: The 4 bits of status character 0 for the previously selected RAM register are transferred to the accumulator. The carry/link and the status character are unaffected.

Mnemonic: RD1 (Read RAM status character 1)
OPR OPA: 1110 1101
Symbolic: $(M_{S1}) \rightarrow ACC$

Mnemonic: RD2 (Read RAM status character 2)
OPR OPA: 1110 1110
Symbolic: $(M_{S2}) \rightarrow ACC$

Mnemonic: RD3 (Read RAM status character 3)
OPR OPA: 1110 1111
Symbolic: $(M_{S3}) \rightarrow ACC$

Mnemonic: RDR (Read ROM port)
 OPR OPA: 1110 1010
 Symbolic: (ROM input lines) → ACC
 Description: The data present at the input lines of the previously selected ROM chip is transferred to the accumulator. The carry/link is unaffected.

If the I/O option has both inputs and outputs within the same 4 I/O lines, the user can choose to have either "0" or "1" transferred to the accumulator for those I/O pins coded as outputs, when an RDR instruction is executed.

Example: Given a port with I/O coded with 2 inputs and 2 outputs, when RDR is executed the transfer is as shown below:

I ₃	O ₂	O ₁	I ₀		(ACC)
1	X	X	0	⇒	1 (1 or 0) (1 or 0) 0
↑		↑			↑ ↑
Input Data					User can choose

Mnemonic: WRM (Write accumulator into RAM character)
 OPR OPA: 1110 0000
 Symbolic: (ACC) → M
 Description: The accumulator content is written into the previously selected RAM main memory character location. The accumulator and carry/link are unaffected.

Mnemonic: WRO (Write accumulator into RAM status character 0)
 OPR OPA: 1110 0100
 Symbolic: (ACC) → M_{S0}
 Description: The content of the accumulator is written into the RAM status character 0 of the previously selected RAM register. The accumulator and the carry/link are unaffected.

Mnemonic: WR1 (Write accumulator into RAM status character 1)
 OPR OPA: 1110 0101
 Symbolic: (ACC) → M_{S1}

Mnemonic: WR2 (Write accumulator into RAM status character 2)
 OPR OPA: 1110 0110
 Symbolic: (ACC) → M_{S2}

Mnemonic: WR3 (Write accumulator into RAM status character 3)
 OPR OPA: 1110 0111
 Symbolic: (ACC) → M_{S3}

Mnemonic: WRR (Write ROM port)
 OPR OPA: 1110 0010
 Symbolic: (ACC) → ROM output lines
 Description: The content of the accumulator is transferred to the ROM output port of the previously selected ROM chip. The data is available on the output pins until a new WRR is executed on the same chip. The ACC content and carry/link are unaffected. (The LSB bit of the accumulator appears on I/O₀.) No operation is performed on I/O lines coded as inputs.

Mnemonic: WMP (Write memory port)
 OPR OPA: 1110 0001
 Symbolic: (ACC) → RAM output register
 Description: The content of the accumulator is transferred to the RAM output port of the previously selected RAM chip. The data is available on the output pins until a new WMP is executed on the same RAM chip. The content of the ACC and the carry/link are unaffected. (The LSB bit of the accumulator appears on O₀, Pin 16, of the 4002.)

Mnemonic: ADM (Add from memory with carry)
 OPR OPA: 1110 1011
 Symbolic: (M) + (ACC) + (CY) → ACC, CY
 Description: The content of the previously selected RAM main memory character is added to the accumulator with carry. The RAM character is unaffected.

Mnemonic: SBM (Subtract from memory with borrow)
 OPR OPA: 1110 1000
 Symbolic: $(\overline{M}) + (ACC) + (\overline{CY}) \rightarrow ACC, CY$
 Description: The content of the previously selected RAM character is subtracted from the accumulator with borrow. The RAM character is unaffected.

F. Accumulator Group Instructions

Mnemonic: CLB (Clear both)
 OPR OPA: 1111 0000
 Symbolic: 0 → ACC, 0 → CY
 Description: Set accumulator and carry/link to 0.

Mnemonic: CLC (Clear carry)
 OPR OPA: 1111 0001
 Symbolic: 0 → CY
 Description: Set carry/link to 0

Mnemonic: CMC (Complement carry)
 OPR OPA: 1111 0011
 Symbolic: $(\overline{CY}) \rightarrow CY$
 Description: The carry/link content is complemented

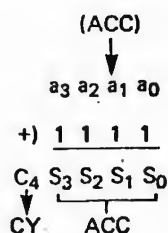
Mnemonic: STC (Set carry)
 OPR OPA: 1111 1010
 Symbolic: 1 → CY
 Description: Set carry/link to a 1

Mnemonic: CMA (Complement Accumulator)
 OPR OPA: 1111 0100
 Symbolic: $a_3 a_2 a_1 a_0 \rightarrow \overline{ACC}$
 Description: The content of the accumulator is complemented. The carry/link is unaffected.

Mnemonic: IAC (Increment accumulator)
 OPR OPA: 1111 0010
 Symbolic: (ACC) + 1 → ACC
 Description: The content of the accumulator is incremented by 1. No overflow sets the carry/link to 0; overflow sets the carry/link to a 1.

Mnemonic: DAC (decrement accumulator)
OPR OPA: 1111 1000
Symbolic: (ACC) - 1 → ACC
Description: The content of the accumulator is decremented by 1. A borrow sets the carry/link to 0, no borrow sets the carry/link to a 1.

Example:



Mnemonic: RAL (Rotate left)
OPR OPA: 1111 0101
Symbolic: $C_0 \rightarrow a_0, a_i \rightarrow a_{i-1}, a_3 \rightarrow CY$
Description: The content of the accumulator and carry/link are rotated left.

Mnemonic: RAR (Rotate right)
OPR OPA: 1111 0110
Symbolic: $a_0 \rightarrow CY, a_i \rightarrow a_{i+1}, C_0 \rightarrow a_3$
Description: The content of the accumulator and carry/link are rotated right.

Mnemonic: TCC (Transmit carry and clear)
OPR OPA: 1111 0111
Symbolic: $0 \rightarrow ACC, (CY) \rightarrow a_0, 0 \rightarrow CY$
Description: The accumulator is cleared. The least significant position of the accumulator is set to the value of the carry/link. The carry/link is set to 0.

Mnemonic: DAA (Decimal adjust accumulator)
OPR OPA: 1111 1011
Symbolic: (ACC) + 0000 → ACC
 or
 0110
Description: The accumulator is incremented by 6 if either the carry/link is 1 or if the accumulator content is greater than 9. The carry/link is set to a 1 if the result generates a carry, otherwise it is unaffected.

Mnemonic: TCS (Transfer carry subtract)
OPR OPA: 1111 1001
Symbolic: 1001 → ACC if (CY) = 0
 1010 → ACC if (CY) = 1
 0 → CY
Description: The accumulator is set to 9 if the carry/link is 0.
 The accumulator is set to 10 if the carry/link is a 1.
 The carry/link is set to 0.

Mnemonic: KBP (Keyboard process)
 OPR OPA: 1111 1100
 Symbolic: (ACC) → KBP ROM → ACC
 Description:

A code conversion is performed on the accumulator content, from 1 out of n to binary code. If the accumulator content has more than one bit on, the accumulator will be set to 15 (to indicate error). The carry/link is unaffected. The conversion table is shown below.

(ACC) before KBP		(ACC) after KBP
0 0 0 0	→	0 0 0 0
0 0 0 1	→	0 0 0 1
0 0 1 0	→	0 0 1 0
0 1 0 0	→	0 0 1 1
1 0 0 0	→	0 1 0 0
0 0 1 1	→	1 1 1 1
0 1 0 1	→	1 1 1 1
0 1 1 0	→	1 1 1 1
0 1 1 1	→	1 1 1 1
1 0 0 1	→	1 1 1 1
1 0 1 0	→	1 1 1 1
1 0 1 1	→	1 1 1 1
1 1 0 0	→	1 1 1 1
1 1 0 1	→	1 1 1 1
1 1 1 0	→	1 1 1 1
1 1 1 1	→	1 1 1 1

Mnemonic: DCL (Designate command line)
 OPR OPA: 1111 1101
 Symbolic: $a_0 \rightarrow CM_0$, $a_1 \rightarrow CM_1$, $a_2 \rightarrow CM_2$
 Description:

The content of the three least significant accumulator bits is transferred to the command control register within the CPU.

This instruction provides RAM bank selection when multiple RAM banks are used. (If no DCL instruction is sent out, RAM Bank number zero is automatically selected after application of at least one RESET). DCL remains latched until it is changed.

The selection is made according to the following truth table.

(ACC)	CM - RAM _i Enabled	Bank No.
X 0 0 0	CM - RAM ₀	Bank 0
X 0 0 1	CM - RAM ₁	Bank 1
X 0 1 0	CM - RAM ₂	Bank 2
X 1 0 0	CM - RAM ₃	Bank 3
X 0 1 1	CM - RAM ₁ , CM - RAM ₂	Bank 4
X 1 0 1	CM - RAM ₁ , CM - RAM ₃	Bank 5
X 1 1 0	CM - RAM ₂ , CM - RAM ₃	Bank 6
X 1 1 1	CM - RAM ₁ , CM - RAM ₂ , CM - RAM ₃	Bank 7

INSTRUCTION	DATA @ X ₂				DATA @ X ₃				COMMENTS
	D ₃	D ₂	D ₁	D ₀	D ₃	D ₂	D ₁	D ₀	
NOP	1	1	1	1	1	1	1	1	
JCN	1	1	1	1	1	1	1	1	
A ₂ , A ₁	1	1	1	1	1	1	1	1	
FIM RRR0	(RRR0)				(RRR1)				The content of address pair RRR
D ₂ D ₁	1	1	1	1	1	1	1	1	
SRC RRR1	(RRR0)				(RRR1)				
FIN RRR0	(RRR0)				(RRR1)				
2nd cycle	1	1	1	1	1	1	1	1	
JIN RRR0	(RRR0)				(RRR1)				
JUN A ₃	A ₃				1 1 1 1				
A ₂ , A ₁	A ₃				1 1 1 1				
JMS A ₃	A ₃				1 1 1 1				
A ₂ , A ₁	A ₃				1 1 1 1				
INC RRRR	(RRRR)				(RRRR) +1				Content of register RRRR; Content +1 of RRRR
ISZ RRRR	(RRRR)				(RRRR) +1				
A ₂ , A ₁	1	1	1	1	1	1	1	1	Content of register RRR
ADD RRRR	(RRRR)				1 1 1 1				
SUB RRRR	(RRRR)				1 1 1 1				
LD RRRR	(RRRR)				1 1 1 1				Content of register RRRR; the content of ACC
XCH RRRR	(RRRR)				(ACC)				
BBL	DDDD				1 1 1 1				
LDM	DDDD				1 1 1 1				Data DDDD
WRM, WR0, WR1, WR2, WR3,	(ACC)				1 1 1 (CY)				Content of accumulator; Content of CY F/F is present on D ₀
WPM, WMP, WRR	(ACC)				1 1 1 (CY)				
RDM, RD0, RD1, RD2 RD3, ADM, SBM, RDR	(M) or (Input)				(M) or (INPUT)				Data fetched from RAM or input
CLB, CLC, IAC, CMC CMA, RAL, PAR, TCC TCS	0	0	0	0	1	1	1	1	
STC, DAC, DCL	1	0	0	1	1	1	1	1	
DAA	0	0	0	0 or 0 1 1 0	1	1	1	1	X ₂ depends on ACC content
KBP	0000,0001,0010 0011,0100,1111				1 1 1 1				X ₂ depends on ACC content

Figure 1-22. 4004 Data Bus Content During Execution of Each Instruction.

INSTRUCTION	DATA @ X ₂				DATA @ X ₃				COMMENTS
	D ₃	D ₂	D ₁	D ₀	D ₃	D ₂	D ₁	D ₀	
NOP	1	1	1	1	1	1	1	1	
HLT*	1	1	1	1	1	1	1	1	After execution of HLT, processor enters STOP mode.
BBS	(SRCH)				(SRCL)				(SRCH) means contents of 4 high order bits of SRC register.
LCR	(COM. REG)				1 1 1 1				
OR4	(0100)				1 1 1 1				
OR5	(0101)				1 1 1 1				
AN6	(0110)				1 1 1 1				
AN7	(0111)				1 1 1 1				
DB0	1	1	1	1	1	1	1	1	
DB1	1	1	1	1	1	1	1	1	
SB0	1	1	1	1	1	1	1	1	
SB1	1	1	1	1	1	1	1	1	
EIN	1	1	1	1	1	1	1	1	
DIN	1	1	1	1	1	1	1	1	
RPM	(P.M.)				(P.M.)				Program memory content

Figure 1-23. Summary of 4040 Data Bus Content During Instruction Execution.

CHAPTER 2 INTRODUCTION TO PROGRAMMING THE MCS-40™ MICROCOMPUTER SYSTEM

PROGRAMMING
THE MCS-40

Writing sequences of instructions for a computer is known as programming. To be able to program a computer effectively, the programmer must understand the action of each of the machine instructions. (The instruction set of the MCS-40 microcomputer is described in detail in the section.)

Each machine instruction manipulates data in some way. The data may be the contents of the program counter which indicates where the next instruction is to be found, the contents of one of the CPU registers, accumulator, or carry flip-flop, the contents of RAM or ROM, or the signals at a port.

Programming is probably most easily learned by use of examples. In the pages that follow, a number of sample program segments are described. In general, the examples are shown in order of increasing complexity. These examples have been chosen to illustrate the use of the I/O ports, basic program loops, multiple precision arithmetic, and the use of subroutines. When reviewing these examples refer frequently to the instruction definitions.

Example #1

Consider the case where it is desired to test the status of a single switch connected to the CPU (4004 chip) on the test input (pin 10). A jump on condition instruction (JCN) can be used to perform this test. Suppose the JCN instruction: JCN TEST, 16 (2 word instruction) is stored at ROM memory locations 2 and 3. The instruction would look as follows:

	OPR	OPA
		C ₁ C ₂ C ₃ C ₄
Location #2	0001 (JCN)	0 0 0 1 (Jump if test signal = Logic "0")
Location #3	0001 (Jump to ROM memory Location #16)	0 0 0 0

When this instruction is executed, if the switch connects a logic "0" (ground) to the test pin of the CPU, the program counter in the address register in the CPU will jump to 16.

(That is, the next instruction to be executed would be fetched from ROM Memory location 16.) If the switch had been connected to a logic "1" (negative voltage) the program counter would not jump but would be incremented by 1 and hence the instruction in ROM memory location 4 would be executed next. Thus the switch status can be tested simply with one instruction. Furthermore, if it were desired to jump if a test signal equalled a logic "1", the JCN instruction could be coded

	OPR	OPA
		C ₁ C ₂ C ₃ C ₄
Location #2	0001	1 0 0 1 Inverted jump condition
Location #3	0001	0 0 0 0

In this case the invert condition bit C₁ is used to indicate a jump is to be made on a logic "1" on the test signal.

If more switches are required a ROM port may be used as shown in the next example.

Example #2

Consider the case where it is desired to test the status of a switch connected to the port of ROM #2. To make access to the port, it is necessary to execute an SRC instruction. The SRC instruction utilizes the contents of a pair of registers, which must contain the proper numbers to select the desired port. Register pairs may be most easily loaded using the FIM instruction.

Thus the sequence

Mnemonic	Description
FIM 0, 20H	Fetch immediate (direct) from ROM data (0010, 0000), to index register pair 0. (20H refers to 20 Hexi-decimal.)
SRC 0	Send the contents of index register pair 0 to select a ROM. The first 4 bits of data sent out at X ₂ time (0010) select ROM #2 (4001).

RDR ;Read the contents of the previously selected ROM (ROM #2) input port into the accumulator.

has the effect of loading the accumulator with the values appearing at ROM port #2. Individual bits may be tested by shifting them into the carry flip-flop and using a jump on condition instruction. In this manner up to 4 switches can be interrogated from one set of ROM input ports (4 of them).

NOTE: All command statements are punctuated with ; at the initial statement Description entry.

Example #3

Suppose a series of 10 clock pulses must be generated, perhaps to drive the clock line of a 4003 port expander. Let us assume that RAM #3 is to be used. The high order 2 bits of data sent out at X₂ time during an SRC instruction selects the RAM chip. Hence 1100 (binary equivalent of 12) is required at X₂ to select RAM #3.

Since we must select the port on RAM #3 we will require

FIM 0,0COH;192

SRC 0

This pair of instructions sets up the desired port for use. To generate the clock pulses, we must alternately write a 1 and a 0 into the appropriate port bit. Let us assume that we will only use the high order bit of the port on RAM #3 and that it is initially set at zero (so that the program does not have to reset it). Furthermore, let us assume that we do not care about the other three bits of the port.

First let us set the accumulator to 0

LDM 0 ;Set accumulator to 0

We may then complement the high order bit of the accumulator by the sequence

RAL ;Rotate left (accumulator and carry)
CMC ;Complement carry
RAR ;Rotate right (accumulator and carry)

which achieves the operation by shifting the bit into the carry flip-flop, complementing it, and shifting it back.

An alternate way to complement the high order bit is to add 8 (binary 1000) to the accumulator. We may set the

contents of one register, say register 15, to 8 by the sequence:

LDM 8 ;Load data DDDD (1000) to the accumulator.

XCH 15 ;Exchange contents of index register 15 and accumulator

LDM 0 ;Load (0000) to accumulator

The first instruction loads the binary number 1000 into the accumulator and the second places the contents of the accumulator into register 15. Since the prior contents of register 15 are also placed in the accumulator, an LDM instruction is then executed to clear the accumulator.

Now the operation ADD 15 will add the binary value 1000 to the accumulator, because Register 15 contains the value 8.

Note the difference in how the LDM and the XCH and ADD instructions utilize the second half of the instruction. The LDM loads the accumulator with the value carried by the instruction, i.e., in binary code LDM 8 appears as 1101 1000 and loads the accumulator with 1000. However, the ADD and XCH select a register, and the contents of the register are used as data. That is, ADD 8 would add the contents of register 8 to the accumulator, not the value 8.

To generate the sequence of 10 clock pulses, one could repeat the following 4 instructions 10 times.

ADD 15	;Add contents of register 15 (1000 previously stored in the register) to accumulator	} one clock pulse generated
WMP	;Write the contents of the accumulator into the previously selected RAM output port	
ADD 15		
WMP		

However, this would take some 40 instructions. The indexing operation available with the ISZ instruction allows a program loop to be repeated 10 times.

The ISZ instruction increments a selected register. If the register initially contained any value other than the value 15 (binary 1111) the instruction performs a JUMP to an address specified by the instruction. This address must be on the same page (within the same ROM) as the instruction immediately following the ISZ.

If however, the register originally contained 15, the CPU will proceed to execute the next instruction in sequence.

By loading a register, say register 14, with the value 6, on the 10th execution of an ISZ, the processor will proceed to the next instruction in sequence rather than jump.

Execution of the ISZ does not affect the accumulator, so that the accumulator does not have to be "saved" prior to its execution.

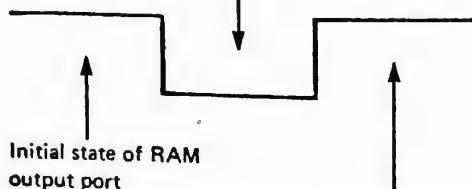
The program sequence which performs the desired action is then

Instruction #	Address Name	Mnemonic	OPA	Description
(1)		LDM	8	;Load 1000 to accumulator
(2)		XCH	15	;Exchange contents of index register 15 and accumulator
(3)		LDM	6	;Load 0110 to accumulator
(4)		XCH	14	;Exchange contents of index register 14 and accumulator
(5)		FIM	0	;Fetch immediate from ROM, Data (1100 0000) to index
		12,	0	;register pair location 0
(6)		SRC	0	;Send address (contents of index register pair 0) to RAM
(7)		LDM	0	;Set accumulator to 0
(8)	→ LOOP	ADD	15	;Add contents of register 15 to accumulator
(9)		WMP		;Write contents of accumulator into RAM output ports
(10)		ADD	15	;Add contents of Register 15 to accumulator
(11)		WMP		;Write contents of accumulator into RAM output ports
(12)		ISZ	14, LOOP	;Increment contents of register 14. Go to ROM address ;A ₂ , A ₁ (called Loop) if result ≠ 0, otherwise skip.

Explanation of Program

- (a) Instruction #1 and #2 – Loads the number 8 (1000) into index register number 15 (1111)
- (b) Instruction #3 and #4 – Loads the number 6 (0110) into index register number 14 (1110)
- (c) Instruction #5 – Fetches the address of the desired RAM and stores it in an index register pair
- (d) Instruction #6 – Sends the stored address to the RAM bank and selects the desired RAM
- (e) Instruction #7 – Initializes the accumulator to 0000.
- (f) Instruction #8, 9, 10, and 11 – Generates one clock pulse as follows:

Complement of highest order bit of accumulator and send back to RAM output port (Instruction #8 and 9)



Highest order bit of accumulator is complemented again and sent back to the RAM output port (Instructions 10 and 11)

(g) Instruction #12

– The contents of Register 14 are incremented by 1 (0001). The number 7 (0111) is now stored in register 14. Since this result is not equal to zero, program control jumps to the address specified in the 2nd word of this instruction. In this case the address stored in the 2nd word is the address of instruction #8. The program then executes the next 4 instructions in sequence and generates a 2nd clock pulse. This sequence is repeated a total of 10 times, thus generating 10 clock pulses. On the 10th time when the contents of register 14 are incremented it goes to the value 0000 and the program skips to the next instruction in sequence and gets out of the loop.

Example #4

Clock pulse streams of the type derived above are often used to drive groups of 4003 shift registers. It may often be desirable to transfer the contents of a RAM register to a group of 4 shift registers via two output ports.

To operate this system, it is necessary to fetch a character from RAM and present it at port #2, then issue the clock pulse at port #1. This sequence requires three SRC commands, one for the RAM selection, one for port #1 selection, and one for port #2 selection.

In addition, the location in RAM must be incremented each time to provide selection of the next character.

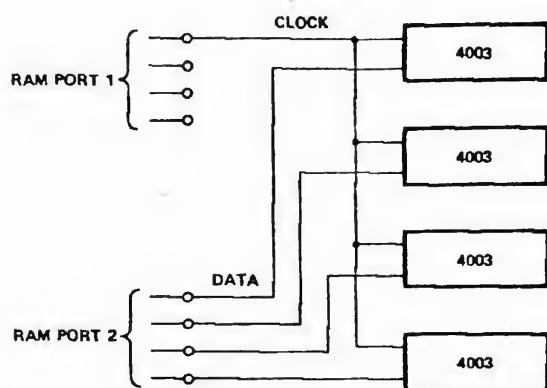


Figure 2-1. RAM Output Ports Driving Groups of Shift Registers

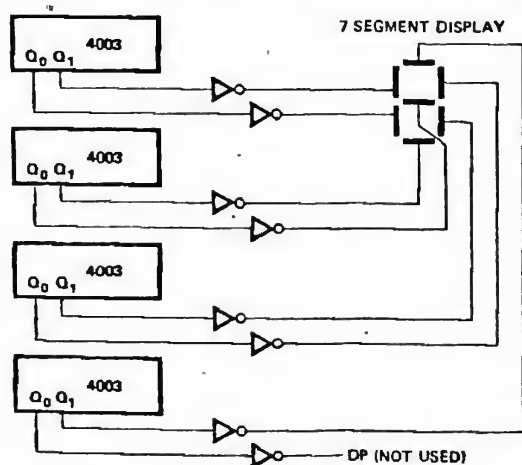


Figure 2-2. Shift Registers Driving Seven Segment LED Displays

The main loop is then as follows:

LOOP:

```
SRC      ;Send address to selected RAM
RDM      ;Read selected RAM character into
          accumulator
SRC      ;Send address to RAM #2
```

```
WMP      ;Write contents of accumulator (previously
          ;selected RAM character) into Port #2
SRC
LDM      0      ;Set accumulator to "0"
ADD      15
WMP      ;Generate 1 clock pulse
ADD      15
WMP
INC      ;Increment by 1 the contents of the register
          ;pair holding the selected RAM address
ISZ      14,LOOP;Increment contents of register 1110.
          ;Jump if result ≠ 0, otherwise skip.
```

The loop above uses 3 pairs of registers for RAM and port selection, and two registers for temporary storage and indexing. The initialization must provide for loading each of these registers.

Example #5

The example above might be extended if for example, the 4003's were driving seven segment LED displays: A 4 line to 7 segment code converter could be used for each display device driven. However, the ROM table lookup capability of the 4040 can be utilized to advantage to save these converters. Suppose the LED displays are wired as shown with each LED using two adjacent locations in each of the 4003's.

The instruction FIN allows a ROM table to be accessed based on the contents of registers 0 and 1. To save register space, the fetched data may be loaded over the table addresses. The table address may be initialized by an FIM or by the sequence

```
LDM
XCH
```

where the data in the LDM represents the high-order 4 bits of the table address. The low order 4 bits will be derived from the data character itself.

The main loop now becomes as follows:

```
FIM      0,DBGH ;initial table address
SRC      ;fetch data character
RDM      ;Read into ACC
XCH      1      ;store at register 1
FIN      0      ;fetch from ROM table
SRC      ;select output port
XCH      0      ;fetch 1st half of 7 segments
WMP      ;transfer to output port
SRC      ;select clock port
LDM      0      ;Set accumulator to "0"
ADD      15
WMP      ;generate one clock pulse
ADD      15
WMP
SRC      ;select output port
XCH      1      ;transfer 2nd half of display
WMP      ;transfer to output port
SRC      ;select clock port
LDM      0      ;Set accumulator to "0"
```

ADD	15	} generate one clock pulse
WMP		
ADD	15	
WMP		
INC		set next RAM character
ISZ	14	test for no. of characters

Note that two data characters (8 bits) are transferred for each digit to be displayed.

This loop must be initialized by setting the registers to their initial conditions. The following sequence of 4 instructions is sufficient:

FIM*	select RAM register for display
FIM	initialize clock port selector
FIM	initialize output port selector
FIM	initialize no. of digits and set reg.

*Operands are device dependent.

Example #6

Proceeding with the example outlined above, suppose that the user finds it necessary to display the contents of a number of different RAM registers, at different places in the program. The sequence of instructions could be used whenever this was necessary. However, by making the entire sequence a "subroutine", the user can call out the sequence each time it's needed with only a JMS instruction.

The JMS utilizes the address push down stack. When a JMS is executed, the program counter is pushed up one level and is reloaded with the address to which the jump to take place, and execution will proceed from this new location. However, before the program counter is reloaded, the old value is saved in the "stack." This stack operates as follows:

1. Each time a JMS is executed, all addresses saved in the stack are pushed down 1 level. The last value of the program counter is loaded into the top of the stack, the program counter value corresponds to the instruction immediately following the JMS.
2. The BBL (BBS in the 4040) instruction raises every entry in the stack one level, with the top value in the stack entering the program counter.

In the example shown, if the RAM register to be transferred to the display is different in different parts of the program, the FIM which selects the RAM register should not be made part of the subroutine. The subroutine would then include the three FIM instructions followed by the main loop and terminated by the BBL or BBS.

To display any register from any point in the program, the programmer need use only 4 bytes of ROM:

FIM	Reg Pair,	Date (Byte)
JMS		

The FIM selects the register and the JMS calls the subroutine.

Example #7

Storing and Fetching a floating point decimal number in the 4002 RAM (How to use the Status and Main Memory Characters in the 4002 RAM)

The 4002 RAM has 4 registers, each with twenty 4 bit characters subdivided into 16 main memory characters and 4 status characters. (320 bits total.) Each register is capable of storing a 20 digit, unsigned, fixed point, binary-coded decimal (BCD) number. A more practical use for the register is the storage of a signed, floating point, BCD number having a 16 digit mantissa (fraction) and a 2 digit exponent.

Consider the number

+	1372994157387406	x 10 ⁻⁵⁹
	Mantissa (16 digits)	Exponent (2 digits)

Storage is required for both the sign of the mantissa (in this case positive) and the sign of the exponent (in this case negative), 16 digits of mantissa and 2 digits of exponent. The 4 status characters of the register can be used to hold the signs (in this case a "1" represents minus - this definition is completely arbitrary and is completely up to the user) and the 2 digit exponent. The 16 main memory characters are used to hold the 16 digit mantissa.

For example, let's store the previously shown number in Bank #2, Chip number #3, register #1. It would be stored in the 4002 as follows:

Register #1					
Decimal digit - 6	0	1	1	0	0
Decimal digit - 0	0	0	0	0	1
Decimal digit - 4	0	1	0	0	2
Decimal digit - 7	0	1	1	1	3
Decimal digit - 8	1	0	0	0	4
Decimal digit - 3	0	0	1	1	5
Decimal digit - 7	0	1	1	1	6
Decimal digit - 5	0	1	0	1	7
Decimal digit - 1	0	0	0	1	8
Decimal digit - 4	0	1	0	0	9
Decimal digit - 9	1	0	0	1	10
Decimal digit - 9	1	0	0	1	11
Decimal digit - 2	0	0	1	0	12
Decimal digit - 7	0	1	1	1	13
Decimal digit - 3	0	0	1	1	14
Decimal digit - 1	0	0	0	1	15
Exponent Value	1	0	0	1	0
59	0	1	0	1	1
Exponent Sign - Neg.	0	0	0	1	2
Mantissa Sign - Pos.	0	0	0	0	3

Main
Memory
Character #

Status
Character #

The following instructions would be used to fetch character #6, the signs, and exponent value:

	Mnemonic	Machine OPR	Language OPA
Select Bank #2	LDM 2	1101	0010
	DCL	1111	1101
Select Chip #3, Register #1 Character #6	FIM 4	0010	1000
	13, 6	11 01	0110
		Chip #3	Main memory Character #8
	SRC 4	0010	1001
Fetch the Mantissa sign From status Character #3 to Register #10 in the CPU	RD3	1110	1111
	XCH 10	1011	1010
Fetch the exponent sign From status Character #2 to Register #11 in the CPU	RD2	1110	1110
	XCH 11	1011	1011
Fetch the exponent from status Character #1 and #0 to Register #12 and #13 respectively	RD1	1110	1101
	SCH 12	1011	1100
	RDO	1110	1100
	SCH 13	1011	1101
Fetch the previously selected main memory Character #6 (which stored the decimal digit 7 to the accumulator	RDM	1110	1001

Example #8 — Interpretive Mode

Interpretive mode programming may be used to reduce the amount of ROM required to implement a particular system function. In this mode, data words fetched from ROM or RAM are treated as instructions of a computer which might be quite different than the MCS-40™ micro-computer. The MCS-40 program "interprets" the data, using it to call appropriate subroutines which simulate the instructions of the different computer. In effect another computer architecture is simulated.

In the interpretive mode, the instructions of the simulated computer (pseudo instructions) may be derived from RAM or ROM. The instructions are fetched from RAM via the normal RAM operations (SRC, RDM), using a simulated program counter to maintain the address. The JIN instruction is often useful for interpreting the fetched instruction. (Address for the JIN is computed from the fetched pseudo

instruction. Each address value is the location of a JMP, or JMS to an appropriate routine, or the routine itself.)

When fetching pseudo instructions from ROM, the FIN is used. As the FIN instruction must be located on the same ROM chip as the fetched data, one cannot use all 256 8 bit bytes of a ROM for pseudo instructions. It is sufficient to allow an FIN followed by a BBL on the ROM chip. Thus up to 254 bytes of each ROM chip can be used for pseudo instructions. The simulated program counter must correspond to this address structure. If the FIN and BBL instructions are located in the first two locations of the ROM chip, the 254 step program address counter can be implemented by initializing the chip address to location 2 rather than location 0. If the interpretive mode program exceeds 254 bytes, the program control routine must determine the proper chip to find the next pseudo instruction. The instruction is then fetched by a JMS to address 0 of the appropriate chip. Refer to the Programmer's Manual for further details.

Example #9 — Interrupt Routine (4040)

The Interrupt signal, when armed and activated, causes the CPU to suspend normal program execution. The CPU is forced to execute a predetermined Interrupt subroutine starting from location 003. At the completion of the Interrupt routine, the CPU is returned to the normal program execution with a BBS instruction.

The Interrupt utilizes the address push down stack. When an Interrupt is executed, the program counter is pushed up one level and is reloaded with address 003. Execution will proceed from this location. The location may contain Jump (JIN, JUN) which allows the user to place the Interrupt routine anywhere in memory. A stack level should always be reserved for the interrupts to avoid overflowing the stack.

Since the Interrupt forces the CPU out of the normal instruction sequence, the state of the CPU's internal register values must be preserved and restored prior to returning from the Interrupt routine. The SRC and the Index Register Bank will automatically be saved. The designer should store the value of the accumulator and carry flip-flop, command register, current ROM bank, and index registers that will be used during the Interrupt execution.

The MCS-40 system has three groups of eight index registers organized into two banks. Registers 8-15 are common to both. Bank 0, Register 0-7 can be designated for normal program execution while Bank 1, Register 0-7 can be designated for interrupt execution. The designer need only switch banks to save the first seven register values. These will be restored automatically with the BBS.

The following is an example of a typical Interrupt subroutine. It illustrates entering and exiting the Interrupt routine, determining an Interrupt source from multiple requests and then directing the Interrupt routine to service the interrupting device.

The programmer must first enable the CPU to accept an Interrupt. This is done by executing the EIN instruction. When an Interrupt occurs, the program counter is forced to location 003 in whichever ROM bank it is executing.

Label	Code	Operand	Comment
Interrupt occurs			
003	JUN	INTR	;Branch to Interrupt
INTR	SB0		;Switch to Bank 0
	XCH	5	;Store accumulator (Acc) in Register (Reg) 5
	TCC		;Place carry (CY) in Acc (a3 bit position)
	XCH	6	;Store carry in Reg 6
	LCR		;Load command Reg (CR) into Acc
	XCH	7	;Store CR in Reg 7
	SB1		;Switch to Bank 1. This bank could typically be used for Interrupt work registers while Bank 0 could be established for normal program execution. This is an arbitrary definition.
TS	JUN	10	;Interrupt vector table
TS+1	JUN	11	;Eight branches, one for each possible source
.	.	.	.
.	.	.	.
.	.	.	.
TS+7	JUN	17	.

Assume there are eight sources of Interrupt. The Interrupt is a common line such that one or more sources can Interrupt simultaneously. The Interrupt Acknowledge should be daisy-chained so that only one device will respond to the Interrupt. This is one technique for arbitrating simultaneous interrupt requests.

Let us further assume that each of the eight sources has a 3 bit binary address. The interrupting device will place this address on ROM 2 (4308) Port 3. This port is designated an input, common to all Interrupt sources.

The program will interrogate the above port, taking the address of the interrupting device and adding it to a table of JUN (TS) instructions. The resulting source will cause the program to vector a unique routine to service the particular interrupting source of interest.

Code	Operand	Comment
FIM	0,176	;Load 1011,0000 into the index register pair 0 (ROM 2, Port 3)
SRC	0	;Select ROM 2, Port 3 for I/O transfer
RDR		;Read in the 3 bit address to Acc. This address will be multiplied by 2 and added to JS. The result will access the JUN of interest.
CLC		;Clear Carry

RAL		;Shift content of Acc toward MSB position (multiple by 2) will allow the TS table to be accessed by 2 (JUN being 2 bytes) on even boundaries.
FIM	8,TS	;Fetch table starting address
CLC		;Clear Carry
ADD	9	;This adds the least significant nibble of TS to the source address.
XCH	1	;This stores the partial sum in Register 1 and loads zero into Acc.
ADD	8	;This adds any previously generated carry into the most significant nibble position of TS.
XCH	0	;Stores the results in Register 0. The indirect address has been accumulated.
JIN	0	;Indirect jump to a table location.

After the unique source interrupt routine has been initiated, a return must be generated. Prior to the return, the previous state of the CPU must be restored.

SB0		;Switch Bank 0
XCH	7	;Read back command register value
*RAL		;Shift a3 (CM-ROM) bit into CY.
*JNC	\$+2	;Jump if CY is zero, hence select DB0 (CM-ROM ₀), otherwise switch banks to DB1 (CM-ROM).
*DB1		;Select CM-ROM.
*RAR		;Shift back.
DCL		;Load command register.
XCH	6	;Fetch carry.
RAR		;Restore carry.
XCH	5	;Fetch accumulator.
BBS		;The program counter is restored to the value prior to the interrupt. The SRC is restored to the current PCL selection. The index register bank selection is restored. The Interrupt Acknowledge line is cleared.

*These instructions may be omitted if only one ROM bank is being used.

The above example assumes that index registers did not contain information to be preserved before or after the Interrupt. If index registers are to be saved, they would be saved in RAM or protected index register zones. This is program dependent.



62
263

106
107

JAZ
LMT

JAZ
BC

14
BC

0 0 LDM 12 JUN 42
1 1 XCH 8 580 44

580 244 LDM 12 DC
1 245 XCH 8 BB
2 246 LDM 15 DF
3 247 XCH 5 B5
4 248 FIM OP 20
5 249 16 10
6 24A SRC OP 21
7 24B JUN 40
581 24C 2 02

2 2 LDM 8 DP
3 3 JMS 50
4 4 IXOP +2 FO
5 5 WRR EZ
6 6 JIZ 11
7 7 * 06
8 8 LD 14 AE
9 9 JAN 1C
10 A * +4 OD
11 B JUN 40
12 C 11
13 D LDM 1314 DE
14 E JUN 42
15 F 589 4D

589 24D JMS 52
90 E RPOFP 11
1 F LDM 5 DS
2 250 SMS 50
3 1 IXOP +2 FO
4 2 LDM 6 8
5 3 JMS 50
6 4 IXOP +2 FO
7 5 JUN 40
588 25L 17 11

64 8055

1-11-79

J.A.P., Notary

Greg Cox X
LVB
9/11/81

Loc Dec	Loc Hex	NAME	NEW Hex	OLD Hex
2	2	LDM 0	D0	
3	3	JMS	52	
4	4	RPOOP	11	
5	5	LDM 0	D0	
6	6	JMS =	50	
7	7	IXOP+2	F0	
8	8	WRR	E2	
9	9	JT2	11	
10	A	*	09	
11	B	LD 14	AE	
12	C	JAN	1C	
13	D	*+8	14	
14	E	LDM 8	D8	
15	F	XCH 0	B0	
16	10	ISZ 0,	70	
17	11	*	10	
18	2	JUN	40	
19	3	*+8	1A	
20	4	LDM 14	DE	
21	5	JMS	52	
22	6	RPOOP	11	
23	7	LDM 0	D0	
24	8	JMS	52	
25	9	RPOOP	11	
26	A	LDM 1	D1	
27	B	WRR	E2	
28	C	CLB	F0	
29	D	JT2	11	
30	E	*+3	20	
31	F	IAC	F2	
32	20	XCH 4	B4	
33	11	FIM 1P,	22	
34	2	8	08	
35	3	LDM 18 15	AD	
36	4	XCH 0	B0	
37	5	ISZ 0,	70	
38	6	*	25	
39	7	LD 4	A4 ✓	
40	8	LAR	F6 ✓	
41	9	LD 7	A7	
42	A	NOP	00	
43	B	NOP	00	
44	C	JUN	42	
45	2D	589	4D	

Loc	Sec	Loc	Hex	Mnemonic	New Hex	Old Hex
589		24D		JCF	12	
90		E		*+3	50	
1		F		LD 9	49	
2		25D		WRR	E2	
3		1		LDM 15	DE DF	
4		2		JMS	52	
5		3		RPOFP	11	
6		4		LDM 0	D0	
7		5		JMS	52	
8		6		RPOFP	11	
9		7		LD 4	A4	
600		8		RAR	F6	
1		9		LD 6	A6	
2		A		JCF	12	
3		B		*+3	5D	
4		C		LD 8	A8	
5		D		WRR	E2	
6		E		LDM 13	DD	
7		F		JMS	52	
8		260		RPOFP	11	
9		1		LDM 0	D0	
610		2		JMS	52	
1		3		RPOFP	11	
2		4		WRR	E2	
3		5		JT 4	19	
4		6		*	65	
5		7		JUN	40	
611		268		46	2E	
236		EC		JUN	42	
237		ED		47	69	
617		269		LDM 8	D8	
8		A		JMS	50	
9		B		IXP1+2	F0	
20		C		JUN	40	
21		26D		PINBAL	02	
224		EO		JUN	42	
226		E1		622	6E	
622		26E		LDM 8	D8	
3		F		JMS	50	
4		270		IXDP 42	F0	
5		1		JUN	40	
626		272		226	E2	

loc Was Is Machine Code
 282 11A JUN 42
 283 11B 628 74
 628 274 JCZ 1A
 9 5 *+ 7A
 30 6 RAR FL6
 2 7 JCZ 1A
 3 8 *+ 7A
 4 9 INC 0 60
 635 27B FIN OP 30
 JIN OP 31

282 120 JUN 42
 283 121 628 74
 284 274 FIN OP 30
 629 27B JIN OP 31

(1A)

282 11A JUN 42
 283 11B 628 74
 628 274 JCZ 1A
 9 5 27A 7A
 30 6 RAR FL6
 2 7 JCZ 7A
 3 8 27A 7A
 4 9 INC 0 60
 5 27B JUN 41
 284 1C
 284 11C FIN OP 30
 285 11D JIN OP 31

(125)

Loc sec	Loc Hex	NAME/MONIC	New Hex	Old Hex
484	1E7	IRET	0B	
488	1E8		21	
489	9		21	
490	A		25	
491	B		25	
492	C		25	
493	D		31	
494	E		38	
495	F		3E	
496	1F0		56	
497	1		50	
498	2		44	
499	3		5C	
500	4		63	
501	5		6A	
502	6		70	
503	1F7		76	
504	8		7A	
5	9		7E	
6	A		99	
7	B		C9	
8	C		85	
9	D		91	
510	1FE		C9	
324	14F	LD 7	A7	
325	5	PAL	F5	
326	C	STC	FA	
327	7	RAR	F6	
328	148	XCH 7	B7	
342	150	LD 6	AL	
8	7	JMS	52	
4	8	SB1	28	
5	9	JUN	42	
346	15A	636	7C	
636	27C	LD 0	A0	
7	D	XCH 6	B6	
18	E	JUN	41	
639	27F	SCN	BE	
340	154	JUN	42	
341	155	636	80	

Loc Dec	Loc Hex	Mnemonic	New Hex	Old Hex
---------	---------	----------	---------	---------

640

642

645

647

646

647

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

280

281

282

1D1

1D9

2E

2F

283

284

285

286

287

129

12A

121

21A

102

103

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

XCH 6

JUN

329

*+19

*+11

JUN

643

LDM 9

JMS

IXP +2

JUN

48

JCP

SCH

JMS

244

JUN

648

LD 8

RAL

JCP

*+4

JUN

LRET

RAL

RAL

JCP

*+4

JUN

LRET

INC 4

LD 5

JUN

260

B6

41

49

E3

E3

42

83

D9

50

F0

40

30

12

BE

52

F4

42

88

A8

F5

12

8E

42

08

F5

F5

1A

94

42

08

64

A5

41

04

Loc	Loc	Hex	Memorize	New Hex	Old Hex
89	BD	JUN	42		
190	BE	664	98		
664	291	J.T.Φ	19		
5	19	*	98		
6	A	JMS	50		
7	B	GMINC	FA		
8	C	JUN	40		
664	29D	191	BF		
192	CD	JUN	40		
193	CI	204	CC		
198	CE	JUN	43		
199	CF	672	00		
672	300	J.T.Φ	19		
8	301	*	00		
8	302	JMS	50		
6	303	GMINC	FA		
6	304	JUN	40		
672	305	200	CB		
202	CA	JMS	50		
203	CB	GMINC	FA		
227	E3	JAN	1C		
228	E4	*+7	EA		
125	4D	128	80		
81	51	*	50		40
42	D	LDM 18	3E		
591	24F	LDM 0	20		
592	250	JUN	43		
593	251	678	06		
678	306	JMS	38		
9	7	IX01+2	F0		
80	P	LDM 0	D0		
1	9	JMS	52		
2	8	RPOPP	1/2		
3	8	JUN	42		
684	30C	594	52		

229	E5	JUN	43
230	E6	685	00
185	300	JMS	52
6	E	MATCH	20
7	F	JMS	50
8	310	IX 6P	EE
8	1	JUN	40
690	312	231	E7

61	3D	JUN	43
62	3E	691	13
63			
691	313	LDM 6	D6
2	4	WRP	E2
3	1	JT2	11
4	6	*+4	19
5	7	JUN	40
6	8	110	6E
7	9	LD 14	AE
8	A	JAZ	14
9	B	*+4	16
700	C	JUN	40
1	D	64	40
2	E	JUN	40
103	31F	PBI	86

64	40	JUN	43
65	41	704	20
704	320	LDM 5	D5
5	1	WRP	E2
6	2	LD 8	A8
7	3	RAR	F6
8	4	JUN	40
09	325	66	42

230	326	JUN	43
231	327	710	20
232	328	711	20
233	329	712	20
234	330	713	20
235	331	714	20
236	332	715	20
237	333	716	20
238	334	717	20
239	335	718	20
240	336	719	20
241	337	720	20
242	338	721	20
243	339	722	20
244	340	723	20
245	341	724	20
246	342	725	20
247	343	726	20
248	344	727	20
249	345	728	20
250	346	729	20
251	347	730	20
252	348	731	20
253	349	732	20
254	350	733	20
255	351	734	20
256	352	735	20
257	353	736	20
258	354	737	20
259	355	738	20
260	356	739	20
261	357	740	20
262	358	741	20
263	359	742	20
264	360	743	20
265	361	744	20
266	362	745	20
267	363	746	20
268	364	747	20
269	365	748	20
270	366	749	20
271	367	750	20
272	368	751	20
273	369	752	20
274	370	753	20
275	371	754	20
276	372	755	20
277	373	756	20
278	374	757	20
279	375	758	20
280	376	759	20
281	377	760	20
282	378	761	20
283	379	762	20
284	380	763	20
285	381	764	20
286	382	765	20
287	383	766	20
288	384	767	20
289	385	768	20
290	386	769	20
291	387	770	20
292	388	771	20
293	389	772	20
294	390	773	20
295	391	774	20
296	392	775	20
297	393	776	20
298	394	777	20
299	395	778	20
300	396	779	20
301	397	780	20
302	398	781	20
303	399	782	20
304	400	783	20
305	401	784	20
306	402	785	20
307	403	786	20
308	404	787	20
309	405	788	20
310	406	789	20
311	407	790	20
312	408	791	20
313	409	792	20
314	410	793	20
315	411	794	20
316	412	795	20
317	413	796	20
318	414	797	20
319	415	798	20
320	416	799	20
321	417	800	20
322	418	801	20
323	419	802	20
324	420	803	20
325	421	804	20
326	422	805	20
327	423	806	20
328	424	807	20
329	425	808	20
330	426	809	20
331	427	810	20
332	428	811	20
333	429	812	20
334	430	813	20
335	431	814	20
336	432	815	20
337	433	816	20
338	434	817	20
339	435	818	20
340	436	819	20
341	437	820	20
342	438	821	20
343	439	822	20
344	440	823	20
345	441	824	20
346	442	825	20
347	443	826	20
348	444	827	20
349	445	828	20
350	446	829	20
351	447	830	20
352	448	831	20
353	449	832	20
354	450	833	20
355	451	834	20
356	452	835	20
357	453	836	20
358	454	837	20
359	455	838	20
360	456	839	20
361	457	840	20
362	458	841	20
363	459	842	20
364	460	843	20
365	461	844	20
366	462	845	20
367	463	846	20
368	464	847	20
369	465	848	20
370	466	849	20
371	467	850	20
372	468	851	20
373	469	852	20
374	470	853	20
375	471	854	20
376	472	855	20
377	473	856	20
378	474	857	20
379	475	858	20
380	476	859	20
381	477	860	20
382	478	861	20
383	479	862	20
384	480	863	20
385	481	864	20
386	482	865	20
387	483	866	20
388	484	867	20
389	485	868	20
390	486	869	20
391	487	870	20
392	488	871	20
393	489	872	20
394	490	873	20
395	491	874	20
396	492	875	20
397	493	876	20
398	494	877	20
399	495	878	20
400	496	879	20
401	497	880	20
402	498	881	20
403	499	882	20
404	500	883	20
405	501	884	20
406	502	885	20
407	503	886	20
408	504	887	20
409	505	888	20
410	506	889	20
411	507	890	20
412	508	891	20
413	509	892	20
414	510	893	20
415	511	894	20
416	512	895	20
417	513	896	20
418	514	897	20
419	515	898	20
420	516	899	20
421	517	900	20
422	518	901	20
423	519	902	20
424	520	903	20
425	521	904	20
426	522	905	20
427	523	906	20
428	524	907	20
429	525	908	20
430	526	909	20
431	527	910	20
432	528	911	20
433	529	912	20
434	530	913	20
435	531	914	20
436	532	915	20
437	533	916	20
438	534	917	20
439	535	918	20
440	536	919	20
441	537	920	20
442	538	921	20
443	539	922	20
444	540	923	20
445	541	924	20
446	542	925	20
447	543	926	20
448	544	927	20
449	545	928	20
450	546	929	20
451	547	930	20
452	548	931	20
453	549	932	20
454	550	933	20
455	551	934	20
456	552	935	20
457	553	936	20
458	554	937	20
459	555	938	20
460	556	939	20
461	557	940	20
462	558	941	20
463	559	942	20
464	560	943	20
465	561	944	20
466	562	945	20
467	563	946	20
468	564	947	20
469	565	948	20
470	566	949	20
471	567	950	20
472	568	951	20
473	569	952	20
474	570	953	20
475	571	954	20
476	572	955	20
477	573	956	20
478	574	957	20
479	575	958	20
480	576	959	20
481	577	960	20
482	578	961	20
483	579	962	20
484	580	963	20
485	581	964	20
486	582	965	20
487	583	966	20
488	584	967	20
489	585	968	20
490	586	969	20
491	587	970	20
492	588	971	

49
50

31
32

NOP
NOP

00
00

466
467

1D2
1D3

JUN
806

43
26

806
7

326
7

LD 13
IAC

AD
F2

8

8

JAA
XCH 13

FB
BD

0

A

LD 13

AD

1

B

JUN 13
468

41
B4

12

32C

73
74

49
4A

JUN
813

43
23

813

32D

DAC
XCH 14

F8
BE

4

E

LD 14

AC

5

F

JUN 14

40
4B

6

330
331

817

331

475

1DB

JUN

43

476

1DC

818

32

818

332

JMS

50

9

3

IXPL+?

F0

20

E

LD 0

AO

1

5

WRR

E2

2

6

JUN

41

473

337

477

DD

470

1D6

JUN

43

471

1D7

824

38

824

338

XCH 0

80

825

9

LD 0

AO

6

A

JUN

41

7

B

472

08

28

33C

139
140

8B
8C

JUN
832

43
40

832

340

LD 14

AE

3

1

JAZ

14

4

2

*+13

4E

5

3

LDM 0

DO

6

4

JMS

50

7

5

IXOP + 2

FO

8

6

LDM 14

DE

9

7

JMS

5,2

890

8

RPOOP

1.1

1

9

LDM 0

DO

2

A

JMS

52

3

B

RPOOP

1.1

4

C

JMS

50

5

D

IXOP + 1

EF

6

E

LD 12

AE

7

F

JMS

52

848

350

LOOP

00

849

351

JUN

40

850

352

141

8D

20

14

NOP

00

21

15

"

00

22

16

"

00

23

17

"

00

24

18

"

00

25

19

"

00

142

8E

JUN

43

143

8F

832

40

139

8B

JMS

52

140

8C

LOOP

00

846

34E

LD 11

AB

850

352

144

90

84 86 54

1. ... + ... - ...

✓ 20. 241	✓ 241 - 245 - 242	P 24
✓ 241	X 241 - 242	P 24
✓ 241	X 241	P 24
✓ 241	✓ 241 - 242 - 243	P 25
✓ 241 - 242	✓ 241 - 242 - 243	P 25
✓ 241	✓ 241	P 25
✓ 241 - 242	✓ 241 - 242 - 243	P 27
✓ 241	X 241	P 27
✓ 241	X 241	P 27
✓ 241 - 242	✓ 241 - 242	P 21
✓ 241	✓ 241	P 36
✓ 241	✓ 241	P 37

1. ... To ... Cox 1

... ..

✓ 2-45 → 589-616
46

F1

✓ 222-227 → 617-621
12

F2

✓ 224-225 → 622-623
13

F3

✓ 226-227 → 624-625
14

F4

✓ 233-234 → 626-627
15

F5

✓ 235-236 → 628-629
16

F6

✓ 475-476 → 818-823
5

F7

✓ 477-478 → 824-829
5

F8

✓ 189-190 → 634-639
5

F9

✓ 192-193 → 204

F10

✓ 198-199 → 672-677
5

F11

✓ 200-201 → 202-203

F12

✓ 207-208 → 209-210

F13

✓ 125

(1)

F14

✓ 71

(4)

F15

Greg Cox X-18
LVG
9/11/81

$$\checkmark 27-370 \rightarrow 175-270$$

P11

$$\checkmark 61-63 \rightarrow 71-72$$

P17

$$\checkmark 61-63 \rightarrow 71-72$$

P18

$$\checkmark 354-328 \rightarrow 354-328$$

P10

$$\checkmark 342-346 \rightarrow 342-346$$

P20

$$\checkmark 626-629 \rightarrow 626-629$$

P21

$$\checkmark 340-341 \rightarrow 640-642$$

P22

$$\checkmark 465 \rightarrow 465$$

P23

$$\checkmark 472 \rightarrow 472$$

P24

$$\checkmark 465-47 \rightarrow 643-647$$

P25

$$\checkmark 297-297 \rightarrow 297-298$$

P26

$$\checkmark 279 \rightarrow 289$$

P27

$$\checkmark 528 \rightarrow 528$$

P28

$$\checkmark 251-259 \rightarrow 648 \geq 663$$

P29

$$\checkmark 282-282 \rightarrow 622-622$$

P30

1 - 1 - 1

71

1

1

134

12

1-6

127

1.2

627

15

NOTE: To Accumulate COK

Armed 51 Inst.

615 - 617

f4

617 - 618

f11

617 - 618

f12

617 - 618

f12

617 - 618

f12

617 - 618

f12

617 - 618

f12

617 - 618

f12

617 - 618

f12

617 - 618

f12

672 - 677

f11

677 - 682 (with 2, 3, 4)

685 - 690

f16

691 - 703

f17

704 - 709

f18

710 - 716

806 - 812

f15

813 - 817

f16

818 - 823

f17

824 - 828

f18

Greg Cox X-19
LVB 9/11/81

NOTE: To Accumulate, Cook

Agreed 51 I part.

835 - 850

846

810

1-1

1-3

1-5

I/

Cook

J

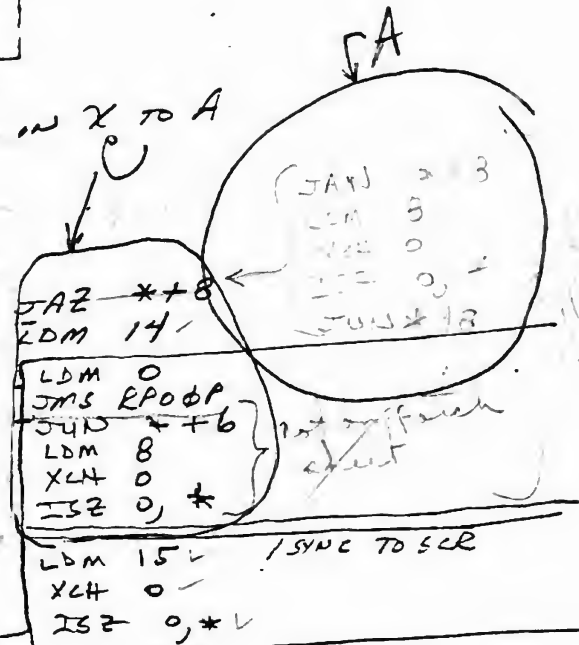
16

NOTE: To Accumulate COK
CHANGE: 265 → NOP
266 → NOP

Added 51 Inst.

0: VECTR = 258
0:
0: / PINBALL EXECUTIVE
0:
0: LDM 12 /SET GAME OVER AND BACKGROUND ON
1: XCH 2
2: PINBAL LDM 0
3: JMS RPOOP
5: ~~LDM 6 /POSITION~~
6: JMS IXOP+2
8: WRR
9: JTZ * //WAIT F
11: LD 14 /ENABLE
12: ~~JAZ +15~~
14: ~~LDM 13~~
15: ~~JMS RPOOP~~
17: LDM 1 /TEST A
18: WRR
19: CLB
20: JTZ *+3
21: IAC
23: XCH 4
24: FIM
26: ~~RAY TO SYNCH SCR TO LIGHTS~~
28: ~~PHASE~~

LDM 15 /SET MISS COUNT
XCH 5
FIM 0, 16 /SELECT ROM 1 FOR I/O
SRC 0



29: ~~CLG~~
30: ~~DAA~~
31: JMS RPOOP /TURN ON LED A SCORE 10,000
33: JMS DLY /DELAY FOR LIGHT
35: LDM 5 /TEST OUT HOLE
36: WRR
37: JTZ PBI
39: CLB /RESET RAI-3, GTB
40: XCH 0
41: LD 14 /TEST BALL COUNT
42: ~~JAZ 0H1~~
44: LD 8
45: RAR
46: JCZ *+6 /TEST SHOOT AGAIN
48: INC 1 /INCREMENT BALL COUNT
49: CLC /RESET SHOOT AGAIN

LDM 0
JMS RPOOP
LD 4
RAR

LDM 0
JMS RPOOP
WRR
JTZ * //WAIT FOR HIT COK
LDM 9 /POSITION X TO J
JMS IXOP+2

LDM 6 /TEST START
WRR
JTZ *+1

JAZ PBI
LDM 5
WRR

Greg Cox X-20
LVB
9/11/81

70: DAC
 71: XCH 8
 72: LD 14 /DECREMENT BALL COUNT
 73: DAC
 74: XCH 14
 75: JAZ OH3
 77: OH2 JMS SOLEN /OUTPUT OUT HOLE KICKER

79: XCH 0
 80: JTN *-1
 82: ISZ 0.*-2

JTN *

84: JUN PINBAL
 86: OH3 LDM 4 /TEST MATCH ENABLE
 87: WRR
 88: JTZ SGO
 90: JMS MATCH /GET MATCH VALUE
 92: CLC
 93: LD 0 /TEST IF A MATCH
 94: SUB 10
 95: JAN SGO
 97: JMS GMING /INCREMENT GAME COUNT
 99: SGO LD 8 /SET GAME OVER LIGHT ON

100: JMS SB2
 102: LD 0
 103: XCH 8
 104: JUN PBI /CONTINUE PROCESSING

~~106: OH1 LDM 6 /TEST START SWITCH~~

~~107: WRR~~

~~108: JTZ PBI~~

~~110: LD 15 /START A NEW GAME~~

PH1 LD 15 /START A NEW GAME

111: JAZ PBI
 113: DAC
 114: XCH 15 /DECREMENT GAMES REMAINING
 115: LDM 7 /TEST 5BALL SWITCH

116: WRR
 117: LDM 3
 118: JTZ **3

120: LDM 5
 121: XCH 14 /SET BALLS REMAINING
 122: FIM 3P,0 /RESET REGISTER DATA
 124: FIM 4P,128

FIM 4P,128

126: FIM 5P,0
 128: FIM 6P,0
 130: LDM 5 /SET ROM PORT TO OUT HOLE KICKER
 131: WRR

132: JUN OH2
 134: PBI CLB /GET ON WITH IT
 135: XCH 4 /RESET HIT COUNT
 136: JMS IXOP /POSITION X TO B
 138: LD 12
 139: JMS LOOP /OUTPUT B LED TEST SWITCHES INCREMENT X
 141: LD 11

142: JMS LOOP /OUTPUT LED C TEST SWITCHES INCREMENT X
 144: LD 10
 145: JMS LOOP /OUTPUT LED D TEST SWITCHES INCREMENT X
 147: LD 14

148: JMS RROOP /OUTPUT LED E
 150: WRR
 151: LD 4 /TEST HIT COUNT

152: JAZ **6
 154: CLB /RESET MISS COUNT
 155: XCH 5
 156: JUN **6

158: ISZ 5.*+4 /INCREMENT MISS COUNT
 160: LDM 15
 161: XCH 5
 162: JTZ *-1

LD 14
 JAZ **6
 LD 0
 JMS 2P,0
 JMS 4P,0
 JMS 6P,0
 LD 11
 JMS 1P,0

164: LD 8 /TURN BACKGROUND OFF

165: RAL

166: CLC

167: RAR

168: XCH 8

169: CLB

170: JUN *+11

172: LDM 1 /TEST TILT

173: WRR

174: JTZ *+11

176: LD 8 /TURN TILT ON

177: JMS SBI

179: LD 0

180: XCH 8

181: CLB /SET BALL COUNT = 0

182: XCH 14

183: JUN SGO /SET GAME OVER

185: LDM 6 /TEST 1P

186: WRR

187: JTZ *+7

189: JMS GMINC /INCREMENT GAME COUNT

191: CLB

192: JUN OH2+2

194: LDM 7 /TEST 3P

195: WRR

196: JTZ *+8

198: JMS GMINC /INCREMENT GAMES REMAINING BY 3

200: JMS GMINC

202: JUN *-13

204: JMS DLY /DELAY FOR LIGHTS

206: JMS IXOP /POSITION X TO F

208: LD 15

209: CLC

210: DAA

211: XCH 4

212: TCC

213: JMS RPOOP /OUTPUT LED F GAMES REMAINING 10

215: JMS DLY /DELAY FOR LIGHTS

217: JMS IXOP /POSITION X TO G

219: LD 4

220: JMS RPOOP /OUTPUT LED G GAMES REMAINING 1

222: JMS DLY /DELAY FOR LIGHTS

224: JMS IXOP /POSITION X TO H

226: LD 14 /TEST BALL COUNT FOR MATCH

227: JAZ *+9

229: JMS MATCH

231: LD 0

232: JMS RPOOP /OUTPUT LED H MATCH TENS

234: JMS DLY /DELAY FOR LIGHTS

236: JUN PINBAL /GO ROUND AGAIN

238: / X COORDINATE OUTPUT ROUTINE

238: /

238: IXOP INC 2 /ENTRY HERE FOR X COORD INCREMENT

239: LD 2

240: WRR /ENTRY HERE FOR X COORD IN ACC

241: LDM 1

242: DCL

243: JUN URET

245: / KNOCKER AND CHIME OUTPUT ROUTINE

245: /

245: KCH WRR /OUTPUT ACC TO DECODER

246: LDM 2

247: DCL /CLOCK DECODER

248: JUN URET

250: / GAME INCREMENT ROUTINE

250: /

JTZ

*+6

JTφ

*

JMS

GMINC

LDM

7

/TEST 3P

WRR

JTZ

*+10

JTφ

*

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

JMS

GMINC

250:GMINC ISZ 15, **4 /INCREMENT GAME COUNT

252: LDM 15

253: XCH 15

254: LDM 3 /OUTPUT KNOCKER

255: JMS KCH

257: BBL 0

258: =512

512:/

512:/ SWITCH TEST LOGIC LOOP

512:/

512:LOOP JMS RPOOP

514: LD 3

515: WRR

516: JTZ **4 /TEST SWITCH

518: JUN VECTR /VECTOR TO SERVICE ROUTINE

520:LRET ISZ 3,LOOP+2

522: LDM 8 /RESET Y

523: XCH 3

524: JMS DLY /DELAY FOR LIGHTS

526: JMS IXOP

528: BBL 0 /RETURN

529:/ ROM PORT 0 OUTPUT ROUTINE

529:/

529:RPOOP FIM OP,0 /SELECT ROM 0

531: SRC OP

532: WRR /OUTPUT ACC TO PORT

533: FIM OP,16 /SELECT ROM 1

535: SRC OP

536: BBL 0 /RETURN

537:/ DELAY ROUTINE

537:/

537:DLY FIM OP,250 /DELAY 1/2 MS

539: ISZ 0,*

541: ISZ 1,*

543: BBL 0

544:/ SCORE MATCH ROUTINE

544:/

544:MATCH CLC /GET A TENS DIGIT MATCH

545: LD 11

546: ADD 12

547: ADD 13

548: CLC

549: DAA

550: XCH 0

551: BBL 0

552:/ SET BIT ROUTINES

552:/

552:SB1 RAR /SET BIT 1

553: RAR

554: STC

555: RAL

556: RAL

557: XCH 0

558: BBL 0

559:SB2 RAL /SET BIT 2

560: RAL

561: STC

562: RAR

563: RAR

564: XCH 0

565: BBL 0

566:/ SOLENOID OUTPUT ROUTINE

566:SOLEN LDM 4 /ISSUE DCL 3-0

567: DCL

568:LRET C/B

569: DCL

244

76144

0: =258
 258:TABLE =488
 258:LRET =520
 258:SOLEN =566
 258:SB1 =552
 258:SB2 =559
 258:GMINC =250
 258:KCH =245
 258:IXOP =238
 258:/SWITCH SERVICE VECTOR ROUTINE
 258:/
 258:VECTR INC 4 /INCREMENT HIT COUNT
 259: LD 5 /TEST MISS COUNT FOR NEW SWITCH
 260: CLC
 261: RAR
 262: JAZ LRET /SAME OLD SWITCH
 264: FIM OP, TABLE /GET JUMP TABLE START ADDRESS
 266: LD 2 /ADD LEAST SIGNIFICANT PART OF OFFSET
 267: RAR
 268: LD 3A
 269: JCZ *+5
 271: RAL
 272: CLC
 273: RAR
 274: CLC
 275: ADD 1
 276: XCH 1
 277: JCZ *+3
 279: INC 0
 280: LD 2 /ADD MOST SIGNIFICANT PART OF OFFSET
 281: RAR
 282: JCZ *+6
 284: RAR
 285: JCZ *+3
 287: INC 0
 288: JIN OP /JUMP TO SERVICE ROUTINE
 289:/SWITCH SERVICE ROUTINES
 289:/
 289:SSHOT JMS SOLEN /SLING SHOTS
 291: JUN SCT
 293:TBUMP JMS SOLEN /THUMPER BUMPERS
 295: LD 3 /TEST IF GREEN OR YELLOW
 296: RAR
 297: JCO SCT
 299: LD 9 /TEST POINT VALUE OF GTB
 300: RAL
 301: JCO SCH
 303: JUN SCT
 305:E LD 7 /E ROLL OVER
 306: RAR
 307: STC
 308: RAL
 309: XCH 7
 310: JUN SCH
 312:L LD 7 /L ROLL OVER
 313: JMS SB1
 315: LD 0
 316: JUN E+4
 318:T LD 7 /T ROLL OVER
 319: JMS SB2
 321: LD 0
 322: JUN E+4
 324:OT LD 8 /OT ROLL OVER

VECTR,

LD 8
 RAL
 JCZ *+4
 JUN LRET
 RAL
 RAL
 JCZ *+4
 JUN LRET
 INC 4 /INCREMENT HIT COUNT

/TEST FOR SLAM & TILT

JAN *+4
 JUN LRET

FIN OP

JCZ SCH

LD 7
 RAL
 STC
 RAR
 XCH 7

321: LD 0
322: XCH 6

329: LD 9 /SET GTB

330: RAL

331: STC

332: RAR

333: XCH 9

334: JUN SCK

336:R LD 6 /R ROLL OVER

337: RAR

338: STC

339: RAL

340: JUN 01+4

342:02 LD 7 /02 ROLL OVER

343: RAL

344: STC

345: RAR

346: JUN E+4

348:MRI LD 9 /MUSHROOM 1

349: RAR

350: STC

351: RAL

352: XCH 9

353: JUN SCK

355:MR2 LD 9 /MUSHROOM 2

356: JMS SB1

358: LD 0

359: XCH 9

360: JUN SCH

362:MR3 LD 9 /MUSHROOM 3

363: JMS SB2

365: LD 0

366: JUN MR1+4

368:RA1 LD 9 /RIGHT ALLEY 1

369: RAR

370: JCO SCK

372: JUN SCH

374:RA2 LD 9 /RIGHT ALLEY 2

375: RAR

376: JUN RA1+1

378:RA3 LD 9 /RIGHT ALLEY 3

379: RAR

380: JUN RA2+1

382:RA4 LD 9 /RIGHT ALLEY RESET

383: RAL

384: LDM 0

385: RAR

386: XCH 9

387: JUN LRET

389:XTRA LD 6 /EXTRA ROLL OVER

390: RAL

391: RAL

392: JCZ SCK

394: LD 8

395: RAR

396: STC

397: RAL

398: XCH 8

399: JUN SCK

401:SPEC LD 6 /SPECIAL ROLL OVER

402: RAL

403: JCZ SCK

405: JMS GMINC

407: JUN SCK

409: / SCORE ROUTINES

409: /

XCH 6
JUN $\phi 1+6$
 $\phi 2$, LD 6
JMS SB1
LD 0
XCH 6
JUN SCH

409: SCT LDM 0 /OUTPUT 10 PT CHIME
 410: JMS KCH
 412: LD 7 /TEST FOR EXTRA & SPECIAL
 413: CMA
 414: JAN **24
 416: LD 6
 417: RAR
 418: JCZ **20
 420: RAR
 421: JCZ **17
 423: LD 10 /TEST EVEN ODD TENS DIGIT
 424: RAR
 425: JCZ **11
 427: LD 8 /TEST SHOOT AGAIN
 428: RAR
 429: LDM 3
 430: JCO **3
 432: LDM 7
 433: XCH 6
 434: JUN **4
 436: LDM 11
 437: XCH 6
 438: LD 10 /INCREMENT SCORE
 439: IAC
 440: DAA
 441: XCH 10
 442: JCO SCH+3
 444: JUN LRET
 446: /

446: SCH LDM 1 /OUTPUT 100 PT CHIME

447: JMS KCH

449: LD 11 /INCREMENT SCORE

450: IAC

451: DAA

452: XCH 11

453: JCO SCH+3

455: JUN LRET

457: /

457: SCK LDM 2 /OUTPUT 1,000 PT CHIME

458: JMS KCH

460: LD 12 /INCREMENT SCORE

461: IAC

462: DAA

463: XCH 12

464: JCZ **20

465: INC 13 /INCREMENT 10,000 DIGIT TEST FREE PLAY

467: LD 13

468: DAC

469: DAC

470: WRR

471: RAL

472: JCO **12

474: LDM 5

475: JMS IXOP+2 /LOOK AT FREE PLAY SWITCHES

477: JTZ **4

479: JMS GMINC /FREE PLAY

481: JMS IXOP+1 /RETURN X TO WHERE IT WAS

483: JUN LRET

485: =488

488: 0+SSHOT /Y=0-7 X=B

489: 0+SSHOT

490: 0+TBUMP

491: 0+TBUMP

492: 0+TBUMP

493: 0+E

LD 13 ✓ / INCREMENT 10,000
 IAC ✓ DIGIT-TEST FREE
 DAA ✓ PLAY
 XCH 13 ✓

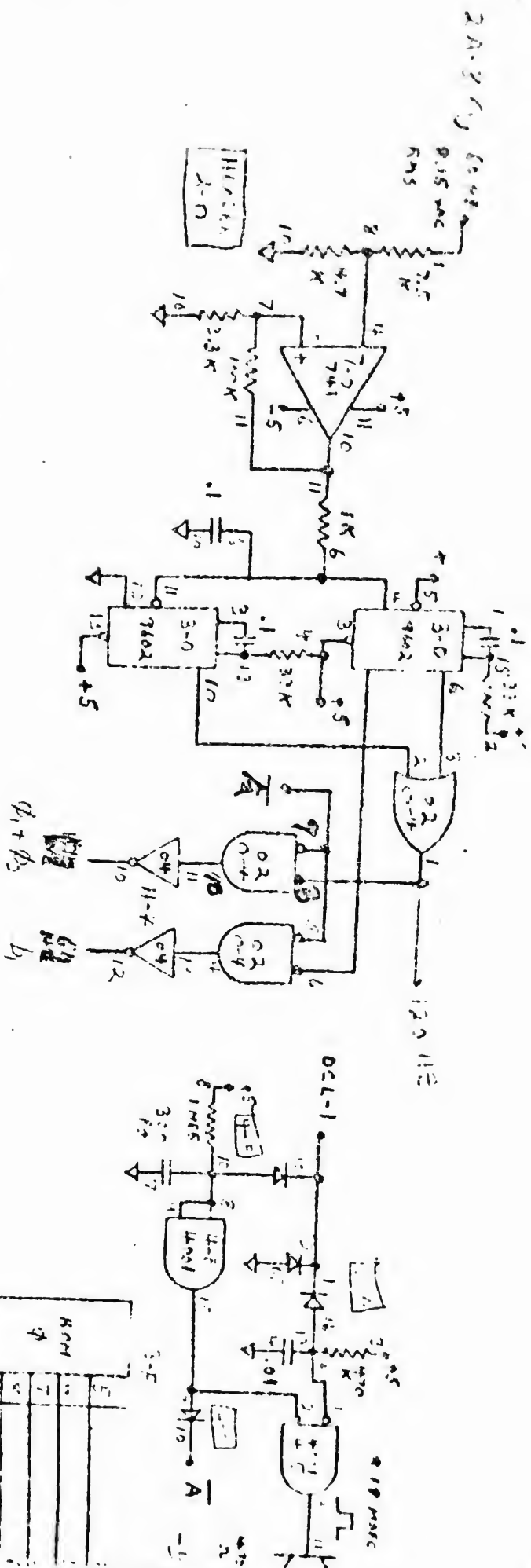
XCH 0 ✓
 LD 0 ✓

LD 0 ✓
 WRR: ✓

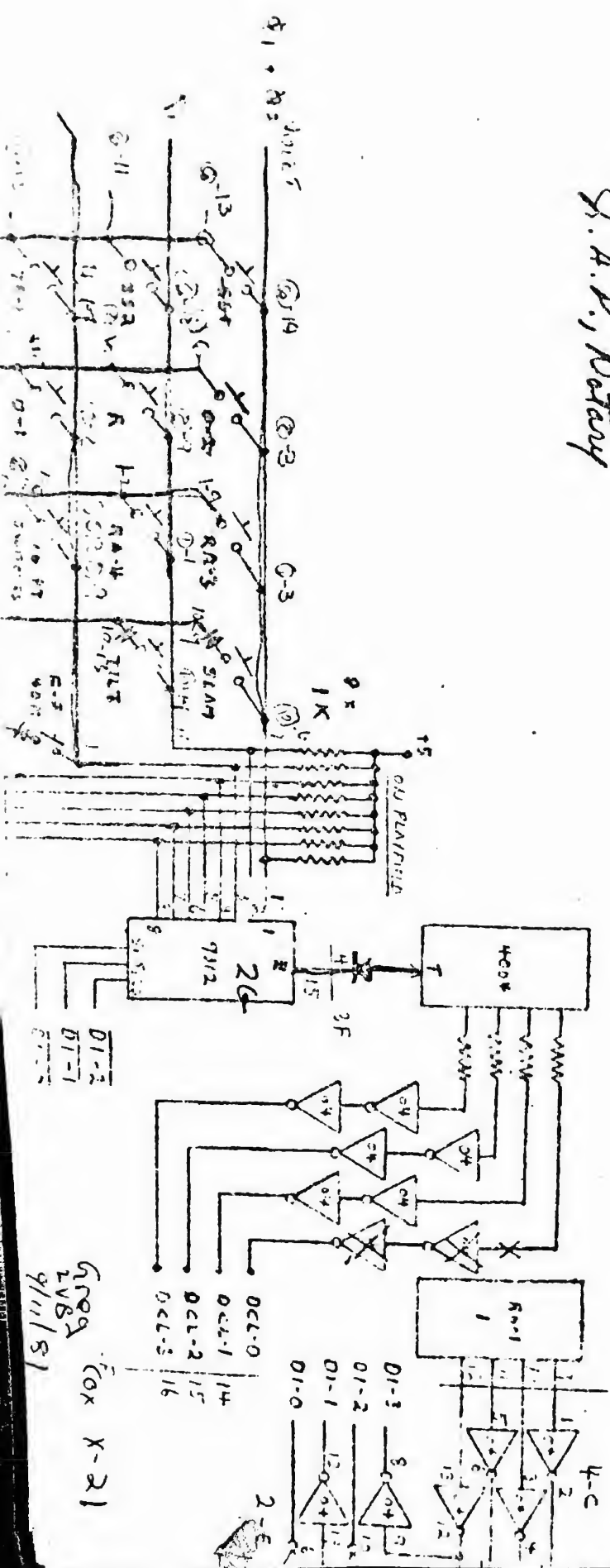
JCZ **26 #19

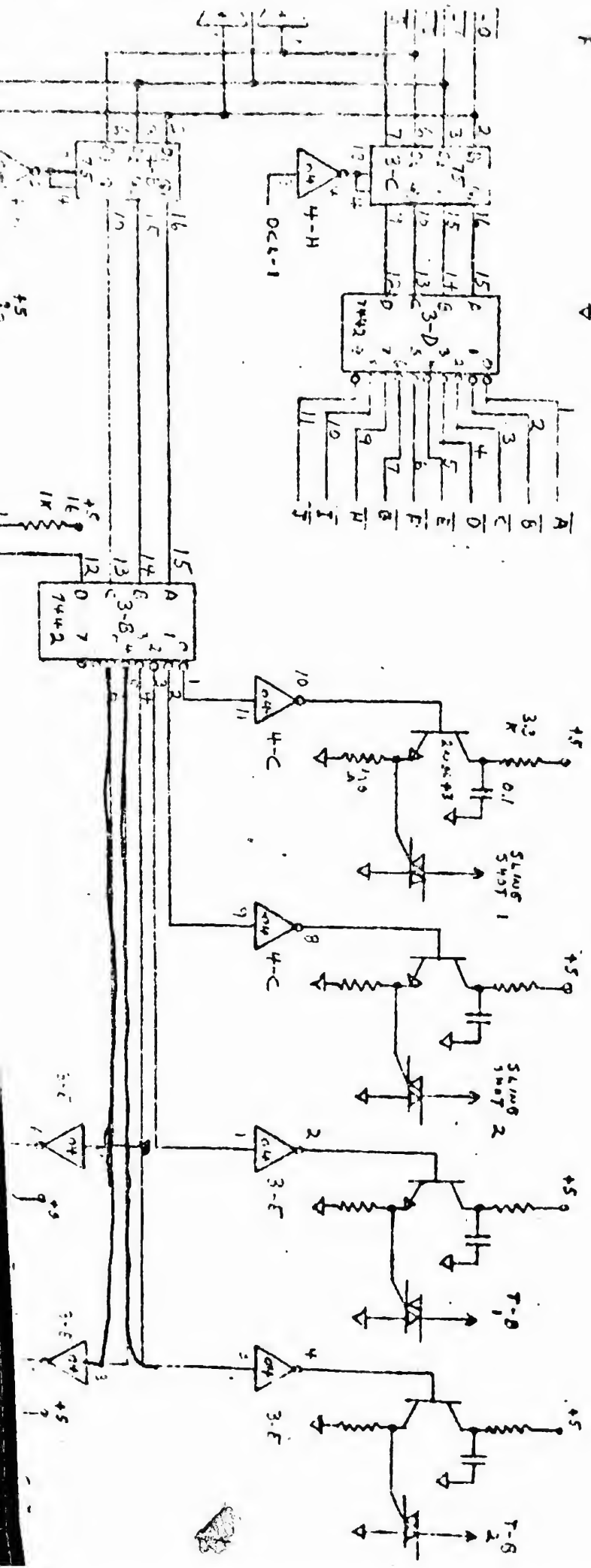
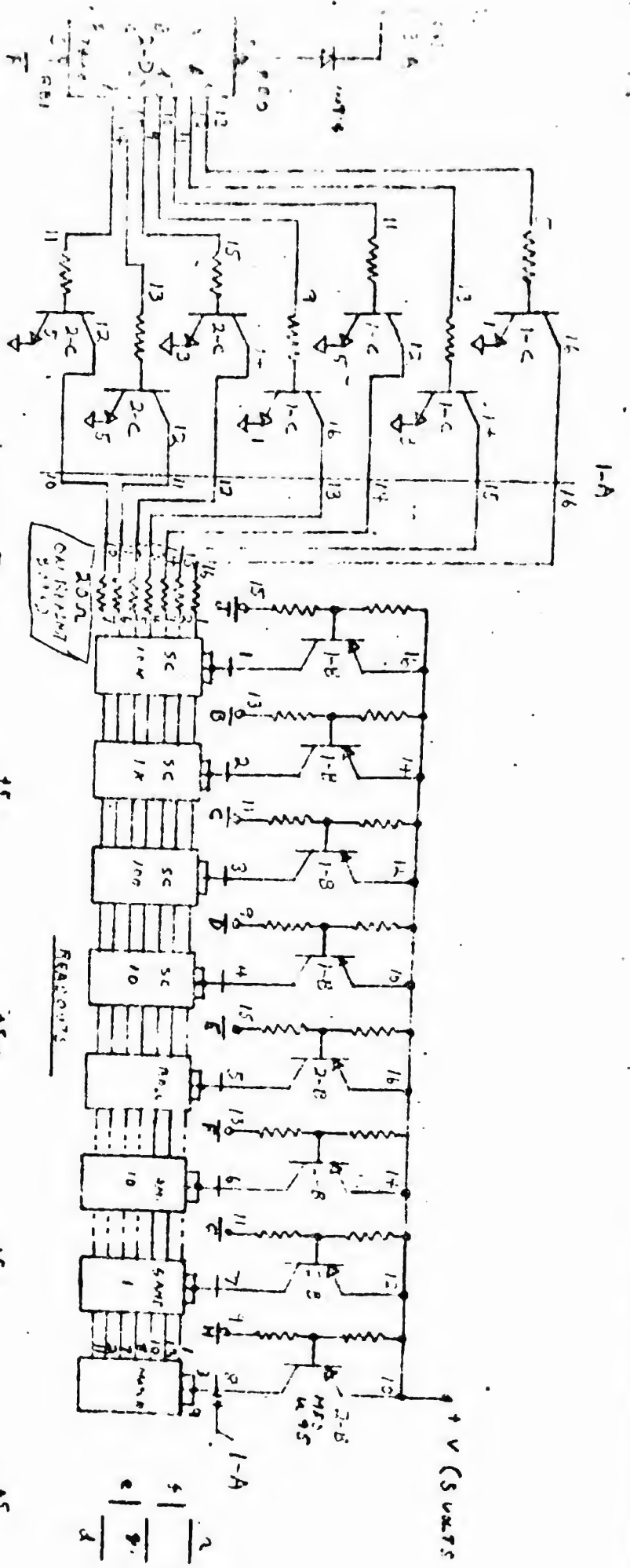
JCO **14

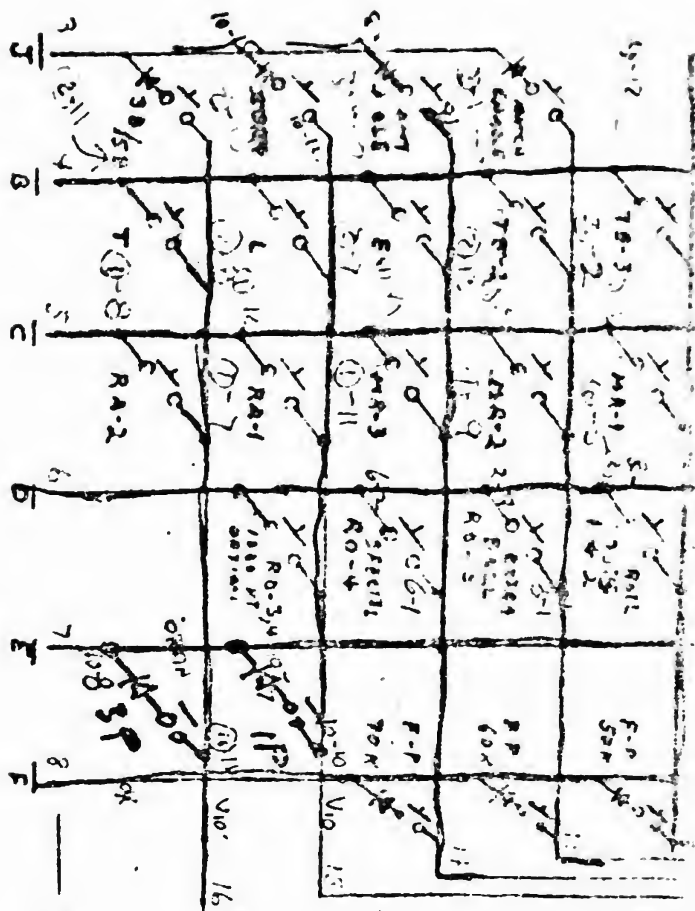
4: 0+L
5: 0+T
6: 0+02 /Y=0-7,X=C
7: 0+R
8: 0+01
9: 0+MR1
00: 0+MR2
01: 0+MR3
02: 0+RA1
03: 0+RA2
04: 0+RA3 /Y=0-6,X=D
05: 0+RA4
06: 0+SCT
07: 0+SCK
08: 0+XTRA
09: 0+SPEC
10: 0+SCK



2A-2 (1)
 8048
 8.5 msc
 RMS
 1-11-79
 S.A.P., Notary





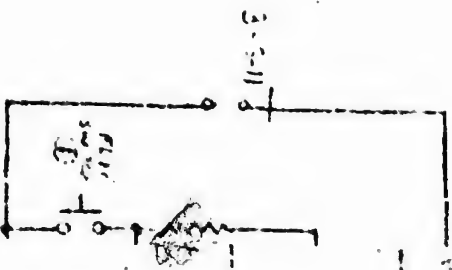


ALL MATRIX DIODES
ARE IN 914

120 HZ

DECL-2

SEVEN SETS
OF
A



P-10-3

EXAMPLE OF THE
POWER SUPPLY CIRCUITRY
USED FOR ALL THE
DUAL-FUNCTION LAMP DRIVERS

LAMPS ARE #47

DIODES ARE 1N4001 (1 AMP, 50 PIV)

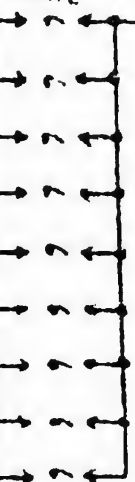
TO SEA
LAMP DRIVER

55 VAC

SOLENOID

TO TAIAC
DRIVER

470K



54

BACKWARD
LAMP

TO BRIDGE
TAIAC

R 0

R 1

R 2

R 3

R 4

R 5

R 6

R 7

R 8

R 9

R 10

R 11

R 12

R 13

R 14

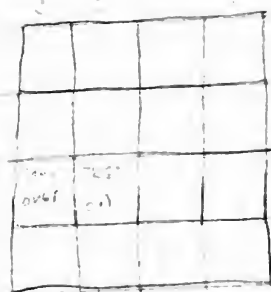
R 15

Loop count (Y) 8-10/4

1 = 1 place

opposite place

1-4 = place
5-8 = place
9-12 = place



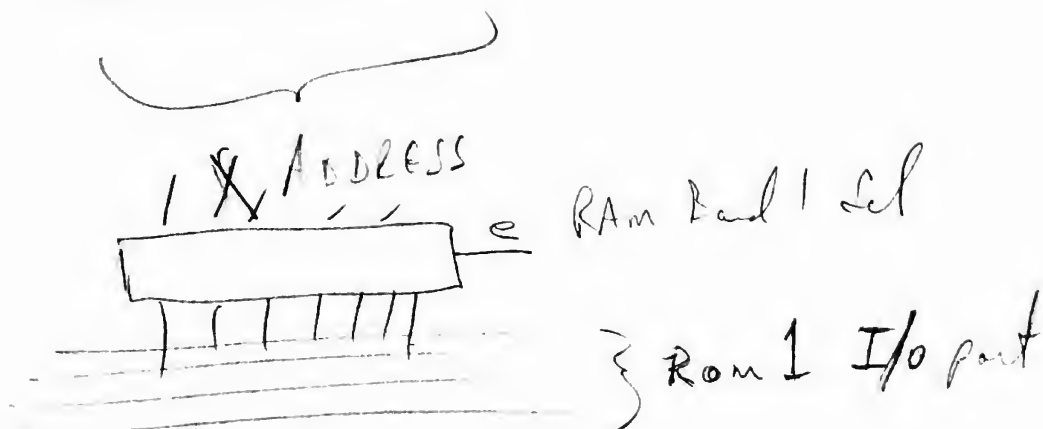
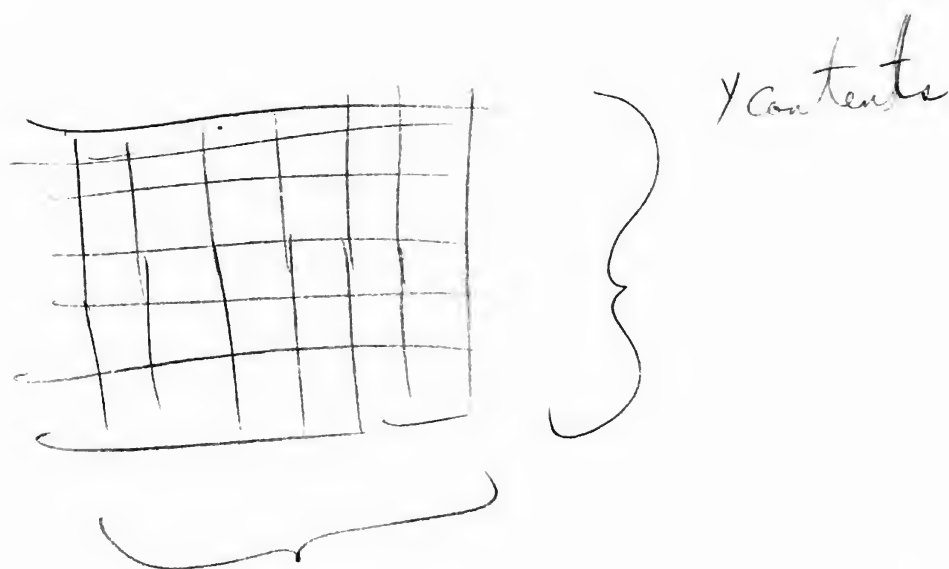
Names Remaining (2 LED's)

1 pt score always 0
10² digit match

Greg Cox X-22A
LvB
9/11/81

can be made

RAM bank 1 should have
as it also the output ROM 1 I/Oing
contents



Greg Cox Y-22B
LVB
9/11/81

X - switch control ROM 1 I/O along to latch using ROM
 Bank select
 J(9) 10,000's LED

- game hangs up if start switch sticks

LED Match row

9	J A	10,000 Score
2 (10)	B	1,000 Score
3 (11)	C	100 Score
4 (12)	D	10 Score
5 (13)	E	Balls remaining
6 (14)	F	games remaining 10's
7 (15)	G	games remaining 1's
8	I	Match 10's

- ~~Red switches not built~~
- game will hang up if switch sticks, i.e. Power switch, TP, etc
- no multiple players - no multiple scores
- Multiple SW closures - multiplexer switch inputs affected all switches inhibit setting any next time if any one set.
 Must be a period of time when no SW set before a switch can be set again.
 i.e. 2 loops thru complete program

Greg Cox X-22C
 LVB
 9/11/81

100 Hz. clock source
 1.0 - 4 0 → 1.4
 7.0 - 1 1 → 1.4

King K. 1000 = 1000 Bank & Out 0-7

turned on/off
 8 weeks data table
 1.0 - 4 0 → 1.4
 7.0 - 1 1 → 1.4
 Seed till

3 play 3 balls each
 1 play 5 balls
 4 Watch enable - allow/disallow free game for watch
 5 Out line select at 1000 start input
 6 Start Switch - increments game count 1 play
 7 Start Switch 3 play / 5 ball SW

8
 9
 10
 11
 12
 13 2nd 4 lights 5-8 enable
 14
 15 1st 4 lights 1-4 enable

16 1st 4 lights 1-4 enable
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536
 537
 538
 539
 540
 541
 542
 543
 544
 545
 546
 547
 548
 549
 550
 551
 552
 553
 554
 555
 556
 557
 558
 559
 560
 561
 562
 563
 564
 565
 566
 567
 568
 569
 570
 571
 572
 573
 574
 575
 576
 577
 578
 579
 580
 581
 582
 583
 584
 585
 586
 587
 588
 589
 590
 591
 592
 593
 594
 595
 596
 597
 598
 599
 600
 601
 602
 603
 604
 605
 606
 607
 608
 609
 610
 611
 612
 613
 614
 615
 616
 617
 618
 619
 620
 621
 622
 623
 624
 625
 626
 627
 628
 629
 630
 631
 632
 633
 634
 635
 636
 637
 638
 639
 640
 641
 642
 643
 644
 645
 646
 647
 648
 649
 650
 651
 652
 653
 654
 655
 656
 657
 658
 659
 660
 661
 662
 663
 664
 665
 666
 667
 668
 669
 670
 671
 672
 673
 674
 675
 676
 677
 678
 679
 680
 681
 682
 683
 684
 685
 686
 687
 688
 689
 690
 691
 692
 693
 694
 695
 696
 697
 698
 699
 700
 701
 702
 703
 704
 705
 706
 707
 708
 709
 710
 711
 712
 713
 714
 715
 716
 717
 718
 719
 720
 721
 722
 723
 724
 725
 726
 727
 728
 729
 730
 731
 732
 733
 734
 735
 736
 737
 738
 739
 740
 741
 742
 743
 744
 745
 746
 747
 748
 749
 750
 751
 752
 753
 754
 755
 756
 757
 758
 759
 760
 761
 762
 763
 764
 765
 766
 767
 768
 769
 770
 771
 772
 773
 774
 775
 776
 777
 778
 779
 780
 781
 782
 783
 784
 785
 786
 787
 788
 789
 790
 791
 792
 793
 794
 795
 796
 797
 798
 799
 800
 801
 802
 803
 804
 805
 806
 807
 808
 809
 810
 811
 812
 813
 814
 815
 816
 817
 818
 819
 820
 821
 822
 823
 824
 825
 826
 827
 828
 829
 830
 831
 832
 833
 834
 835
 836
 837
 838
 839
 840
 841
 842
 843
 844
 845
 846
 847
 848
 849
 850
 851
 852
 853
 854
 855
 856
 857
 858
 859
 860
 861
 862
 863
 864
 865
 866
 867
 868
 869
 870
 871
 872
 873
 874
 875
 876
 877
 878
 879
 880
 881
 882
 883
 884
 885
 886
 887
 888
 889
 890
 891
 892
 893
 894
 895
 896
 897
 898
 899
 900
 901
 902
 903
 904
 905
 906
 907
 908
 909
 910
 911
 912
 913
 914
 915
 916
 917
 918
 919
 920
 921
 922
 923
 924
 925
 926
 927
 928
 929
 930
 931
 932
 933
 934
 935
 936
 937
 938
 939
 940
 941
 942
 943
 944
 945
 946
 947
 948
 949
 950
 951
 952
 953
 954
 955
 956
 957
 958
 959
 960
 961
 962
 963
 964
 965
 966
 967
 968
 969
 970
 971
 972
 973
 974
 975
 976
 977
 978
 979
 980
 981
 982
 983
 984
 985
 986
 987
 988
 989
 990
 991
 992
 993
 994
 995
 996
 997
 998
 999
 1000

Greg Cox
 2/2/83
 9/11/21

KOM d. 1. 1. 1919

LED I/O

negative 20

$$C_A = \frac{1}{2} \left(\frac{1}{2} + \frac{1}{2} \right) = \frac{1}{2}$$

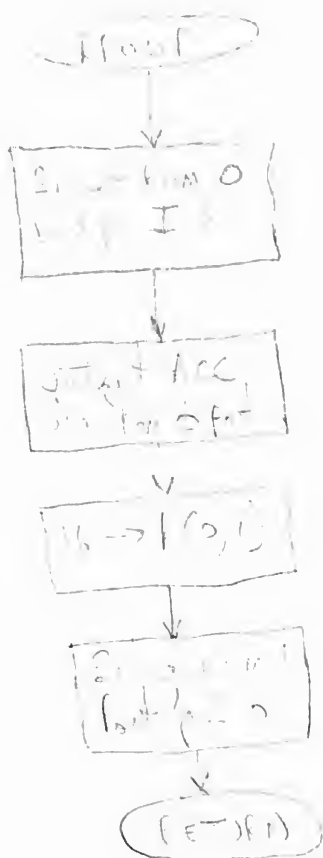
Bank 1 got right

[Handwritten signature]

Greg Cox RRE:

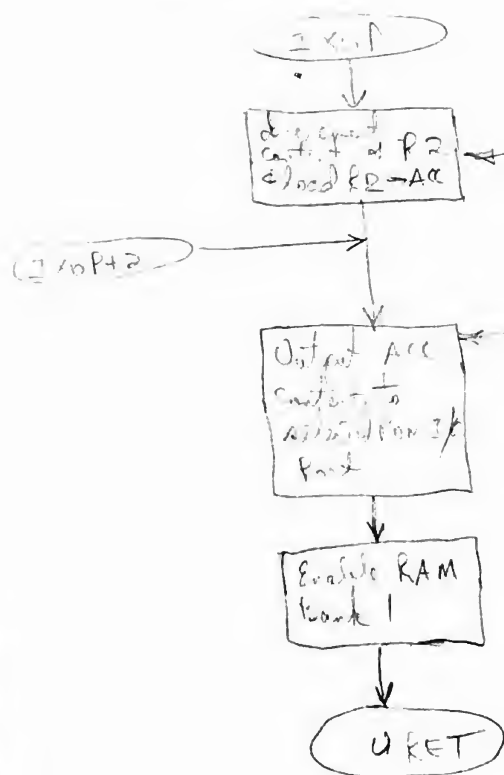
LVB

9/11/81



Form Feed Output

Outputs control of
to form & 1/0 ports

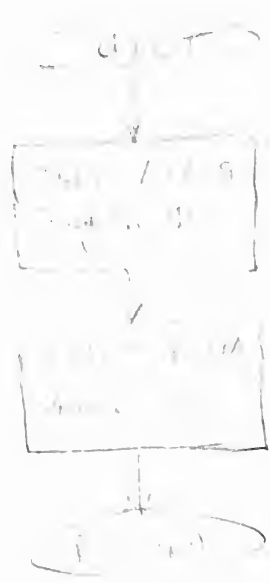


Increment X Coordinate and
Output

entry here for auto increment

entry here for port address

Greg Cox X-22F
LVB
9/11/61



Universal Time

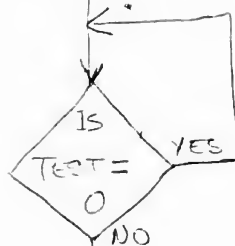
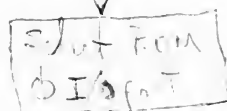
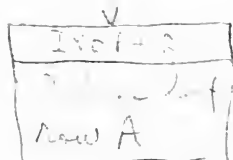
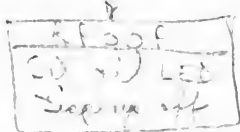
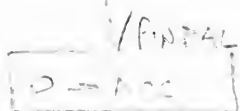
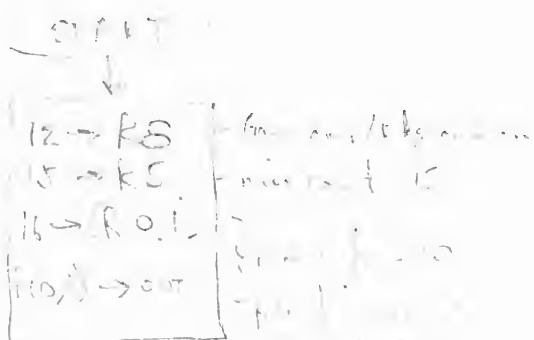
from 1000 hours to 1000
to 1000 hours - 1000
stock

Greg Cox X-226
LVB
9/11/61

From Port ϕ

Greg Cox X-22H
LVB
9/11/81

Loc. 0



Greg Cox X-22J
LvB
9/11/81



Greg Cox X-22K
LVB
9/11/22

U C
1-31-81

AFFIDAVIT OF GREGORY COX

Gregory Cox, being duly sworn, deposes and states as follows:

1. I am currently employed by Kuras-Alterman Corporation which is located at 1573 Route 23, Wayne, New Jersey 07470.

2. From on or about March 4, 1974 until on or about August 16, 1974 I was employed as a computer programmer by Cyan Engineering which was located in Grass Valley, California.

3. I was involved in a project while employed by Cyan Engineering to modify an electromechanical El Toro pinball machine so that the El Toro was controlled by an Intel Intellec development system. The Intellec was connected to an interface board which was cabled to the El Toro by an umbilical cord. Both the Intellec development system and the interface board were located external to the El Toro. As a result of this involvement I became familiar with the design and operation of this modified El Toro.

4. Work on the El Toro project continued from the time that I started at Cyan Engineering until approximately June of 1974. To my knowledge no work was done on this project from about early June of 1974 until I left Cyan Engineering on or about August 16, 1974.

5. For about the first six weeks of my employment at Cyan Engineering there were ongoing discussions concerning our work on the El Toro project between Cyan Engineering personnel and Intel employees and representatives including Intel applications engineers. In these discussions we asked technical questions about the operation of Intel microprocessor related products including the MCS-4 microcomputer chip

Greg Cox X-23
LVB
9/11/81

4C
1/1/81
3/1/81

set and the Intellec development system, and how to interface them to the El Toro and other games. Additionally, such discussions included questions by us about how to program the Intellec development system to control the various components of the El Toro and other games.

on these discussions were used by C.M. & G.C. in the design of the El Toro prototype
6. Steven Mayer, who was also an employee of Cyan Engineering, and *Results and data*
referred in para 5 above
I were involved in these discussions. He participated in most of these circuits.
~~discussions with the Intel people. My participation in these discussions~~
~~included telephone calls from me to Intel applications engineers referred~~
~~by Steven Mayer.~~ *insert (A)*

7. During my conversations I did not disclose to the Intel people any of the details of any work being done at Cyan Engineering. This was consistent with my understanding of a strictly followed policy at Cyan Engineering of not disclosing to outsiders any details of our work. It was my understanding that the reason for this was to prevent disclosure of the details of our work to competitors.

8. I was the person responsible for programming the Intellec development system used with the modified El Toro machine. Within the last month I have reviewed the program which was used in the final stages of the El Toro project, and which bore the designations GD54 and GD55, for the purpose of determining the operation of the El Toro machine as controlled by the Intellec system with such program.

9. My review of the program indicated that there was no provision in the program, and therefore in the system, to act upon and properly respond to the closure of more than one playfield switch simultaneously. As a result, continuous closure of a playfield switch as in the case of a "stuck switch" would prevent proper response of the game to other switches and cause the machine to hang up in a nonoperative state.

10. Also, in this nonoperative state the digital display would not increase the score which normally would have resulted from closure

MC
1/5/81

of switches other than the stuck switch, the solenoids such as the kickers would not actuate, and the lamps would remain in the state they were in prior to the occurrence of the stuck switch.

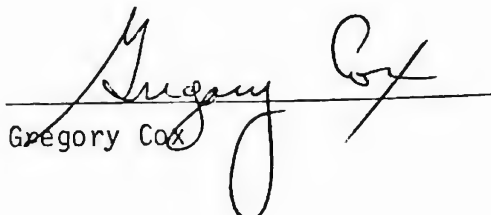
11. To my knowledge no tests were performed on the modified El Toro to determine the machine's susceptibility to electrostatic noise, and no tests were performed on the machine to determine if it accurately scored playfield switch closures other than finger tests which involved actuating the playfield switches with a finger or a ball to see if a proper score was recorded. To my knowledge no tests were performed on the modified El Toro where the response of playfield switch closures to a ball in play were tested in real time other than observation of the game during play.

12. To my understanding, nobody at Cyan Engineering who was in any way involved in the El Toro project had any expertise in the proper operation of a pinball machine.

13. I was present at a Cyan Engineering "open house" which was held on or about June 1974 at the Cyan Engineering facility in Grass Valley, California. Although it was called an "open house" it was not opened to the general public but rather was strictly limited to Atari and Cyan Engineering employees and members of their immediate families. There were approximately 20 people present who spent about two hours viewing the Cyan Engineering facilities, including the laboratory where the El Toro prototype was displayed along with other games. No food or beverages were served at the facilities during the viewing. It was my understanding that everyone present understood that all

MC
3/5/81

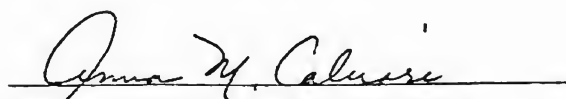
information learned at the "open house" was to be kept strictly company confidential. After the viewing, the participants drove to a park located off the Cyan Engineering facilities for a separate social gathering.



Gregory Cox

(County of Passaic) SS.
(State of New Jersey)

Subscribed and sworn before me this _____ fifth day
of March, 1981.



Notary Public - My Commission
Expires February 9, 1983

MC
3/51

AFFIDAVIT OF GREGORY COX

Gregory Cox, being duly sworn, deposes and states as follows:

1. I am currently employed by Kuras-Alterman Corporation which is located at 1573 Route 23, Wayne, New Jersey 07470.
2. From on or about March 4, 1974 until on or about August 16, 1974 I was employed as a computer programmer by Cyan Engineering which was located in Grass Valley, California.
3. I was involved in a project while employed by Cyan Engineering to modify an electromechanical El Toro pinball machine so that the El Toro was controlled by an Intel Intellec development system. The Intellec was connected to an interface board which was cabled to the El Toro by an umbilical cord. Both the Intellec development system and the interface board were located external to the El Toro. As a result of this involvement I became familiar with the design and operation of this modified El Toro.
4. Work on the El Toro project continued from the time that I started at Cyan Engineering until approximately June of 1974. To my knowledge no work was done on this project from about early June of 1974 until I left Cyan Engineering on or about August 16, 1974.
5. For about the first six weeks of my employment at Cyan Engineering there were ongoing discussions concerning our work on the El Toro project between Cyan Engineering personnel and Intel employees and representatives including Intel applications engineers. In these discussions we asked technical questions about the operation of Intel microprocessor related products including the MCS-4 microcomputer chip

Greg Cox X-24
LVB
9/11/81

set and the Intellec development system, and how to interface them to the El Toro and other games. Additionally, such discussions included questions by us about how to program the Intellec development system to control the various components of the El Toro and other games.

6. Steven Mayer, who was also an employee of Cyan Engineering, and I were involved in these discussions. He participated in most of these discussions with the Intel people. My participation in these discussions included telephone calls from me to Intel applications engineers referred by Steven Mayer.

7. During my conversations I did not disclose to the Intel people any of the details of any work being done at Cyan Engineering. This was consistent with my understanding of a strictly followed policy at Cyan Engineering of not disclosing to outsiders any details of our work. It was my understanding that the reason for this was to prevent disclosure of the details of our work to competitors.

8. I was the person responsible for programming the Intellec development system used with the modified El Toro machine. Within the last month I have reviewed the program which was used in the final stages of the El Toro project, and which bore the designations GD54 and GD55, for the purpose of determining the operation of the El Toro machine as controlled by the Intellec system with such program.

9. My review of the program indicated that there was no provision in the program, and therefore in the system, to act upon and properly respond to the closure of more than one playfield switch simultaneously. As a result, continuous closure of a playfield switch as in the case of a "stuck switch" would prevent proper response of the game to other switches and cause the machine to hang up in a nonoperative state.

10. Also, in this nonoperative state the digital display would not increase the score which normally would have resulted from closure

HC
R
315

of switches other than the stuck switch, the solenoids such as the kickers would not actuate, and the lamps would remain in the state they were in prior to the occurrence of the stuck switch.

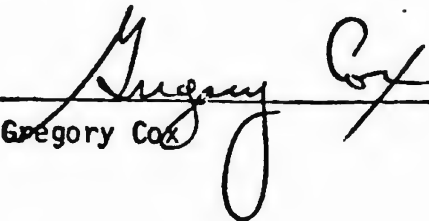
11. To my knowledge no tests were performed on the modified El Toro to determine the machine's susceptibility to electrostatic noise, and no tests were performed on the machine to determine if it accurately scored playfield switch closures other than finger tests which involved actuating the playfield switches with a finger or a ball to see if a proper score was recorded. To my knowledge no tests were performed on the modified El Toro where the response of playfield switch closures to a ball in play were tested in real time other than observation of the game during play.

12. To my understanding, nobody at Cyan Engineering who was in any way involved in the El Toro project had any expertise in the proper operation of a pinball machine.

13. I was present at a Cyan Engineering "open house" which was held on or about June 1974 at the Cyan Engineering facility in Grass Valley, California. Although it was called an "open house" it was not opened to the general public but rather was strictly limited to Atari and Cyan Engineering employees and members of their immediate families. There were approximately 20 people present who spent about two hours viewing the Cyan Engineering facilities, including the laboratory where the El Toro prototype was displayed along with other games. No food or beverages were served at the facilities during the viewing. It was my understanding that everyone present understood that all

HC
3/51

information learned at the "open house" was to be kept strictly company confidential. After the viewing, the participants drove to a park located off the Cyan Engineering facilities for a separate social gathering.

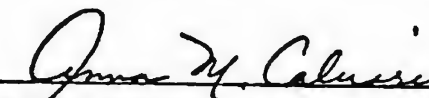


Gregory Cox

(County of Passaic
(State of New Jersey

} SS.

Subscribed and sworn before me this _____ fifth day
of March, 1981.



Notary Public - My Commission
Expires February 9, 1983



Greg Cox X-25
LVB
9/11/81